1970

A One Pass Algorithm for Compiling ALGOL 68 Declarations

Victor B. Schneider

Report Number:
70-053

A ONE-PASS ALGORITHM FOR COMPILING ALGOL 68 DECLARATIONS

Victor B. Schneider

CSD TR 53

October 1970

(Revised October, 1971)

Research for this report was supported
by N.S.F. grant GJ-851
to the Purdue Research Foundation

CONTENTS

	Page
I. Introduction	1
II. Techniques Used in Translation	4
Representation of ALGOL 68 Structures and Data Types in the Compiler	4
Routines Used in Manipulating Compiler List Structures	4
Intermediate Language Generated by the Compiler	8
Preprocessing Implied by Rules of the Grammar	9
Stack Mechanisms Used by Compiler Subroutines	10
Criteria for Selecting Rule Forms in a Translation Grammar	12
III. Translation of Array and Structure Declarations	16
Preliminaries	16
Array Declarations	17
Structure Declarations	21
IV. Translation of Mode Declarations	23
A Final Example	24
Bibliography	26
References Cited in Text	26
APPENDIX 1	27
A Partial Translation Grammer for ALGØL 68	27
APPENDIX 2	37
Translation Rules Written in FORTRAN	37

TABLES

Table		Page
1	List Processing Primitives Used by the Compiler	7

FIGURES

Figure		Page
1	Representation of mode declarations by compiler lists . .	5
1	Continued	6
2	Run-Time Representation of Array Descriptors	19

A ONE-PASS ALGORITHM FOR COMPILING ALGOL 68 DECLARATIONS

I. Introduction

This report describes work underway in the design of an ALGOL 68 compiler system. It presupposes some familiarity with the general ideas of the language, such as might be gained by reading the ALGOL 68 report (1) or some of its companion documents (2,3). In what follows, we are primarily concerned with problems involved in compiling ALGOL 68 data declarations, although mention is made of techniques used in other portions of the compiler as well. Our approach is to explain the motivation behind choosing the translation grammar in Appendix 1 as a means of describing the structure and translation of programs written in the ALGOL 68 language.

The context-free grammar of Appendix 1 was liberally modeled on the Van Wijngaarden grammar of the report (1). That is to say, an attempt was made to force the rules of the grammar to resemble the rules of the report as closely as possible, except in those cases--such as the coercions of Sections 8.2.1 through 8.2.6--where the context-free grammar was clearly not adequate (and is not used) for descriptive purposes. In the case of the rules of Sections 8.2.1 through 8.2.6, the actions represented by rules in the report are to be carried out in the translation rules of the Appendix 1 grammar. Our grammar diverges from the official document principally in minor points, such as the inclusion of rules to describe the so-called "extended language" in Section 9 of the report, and of certain minor syntactic restrictions arising from the one-pass nature of our algorithm, as noted below.

The translation-grammar approach used in this report is based on earlier work by Wirth and Weber (4), Lewis and Stearns (5), Schneider (6,7), and Vere (8). The notation used in this approach can be explained in terms of the following example:

> In the translation grammer of Appendix 1, the rule defining a mode declaration would be written as follows, if we could ignore the table entries and complied code required for such a declaration:
>
> <mode declaration> → (1)
>
> MØDE <mode indicant> = <actual mode declarer>
>
> Rule (1) is exactly the sort of rule one would expect to find in a document such as the ALGOL 60 report (1) which contains a context-free grammar written in B.N.F. In order to describe the process of compiling translated code for a mode declaration, we add information to the context-free rule, as in Appendix 1:
>
> P_{25a}: <mode declaration> →
>
> MØDE/ψ_{25a1}/<mode indicant> = /ψ_{25a2}/
>
> <actual mode declarer>/ψ_{25a3}/
>
> In this expansion of rule (1) above, we have added the notation "P_{25a}:" to indicate that this rule is in the 25th group of rules in our grammar. We have also added three compile-time subroutines, namely, ψ_{25a1}, ψ_{25a2}, and ψ_{25a3}. These compile-time subroutines make entries on compile-time tables and generate code to be written out for the program scanned by the compiler. Thus, when the symbol "MØDE" is recognized by the compiler, subroutine ψ_{25a1} is called; when the sequence of symbols
>
> "MØDE <mode indicant> ="
>
> is recognized by the compiler, subroutine ψ_{25a2} is called; and

the completion of the right-hand side of syntactic rule

P_{25a} causes subroutine ψ_{25a3} to be called.

Thus, the ensemble of rules in a translation grammar may be thought of as an abstract representation of the program input to a compiler-compiler system.

In fact, our translation grammar is intended to be transferred to cards and read in to a compiler-compiler such as the one described in (6). As can be seen in Appendix 2, the subroutines for the data declaration section of our translation grammar have been programmed in FORTRAN V (no less), and it is the operation and interaction of these compiler subroutines that is the main burden of this report.

II. Techniques Used in Translation

Representation of ALGOL 68 Structures and Data Types in the Compiler

ALGOL 68 structures and invented data types are represented as linked lists in the computer memory used by the compiler. Such a representation is, naturally, machine dependent, so we propose here to give examples of how these lists are represented as storage structures on the Purdue C.D.C. 6500 computer. Since the C.D.C. 6000 series of computers has 60-bit memory words, we chose to divide each list word into three address-size fields and one character-size field for storing miscellaneous information. An example follows, showing some typical ALGOL 68 mode declarations and their compile-time representations.

Some typical mode declarations:

```
MØDE $P$  =  PRØC(INT,$S$)REAL;
MØDE $S$  =  STRUCT(REF $S$ PØINTER, $Q$ VALUE);
MØDE $Q$  =  UNIØN($T$, $RS$);
MØDE $RS$ =  REF $S$;
MØDE $T$  =  [1:5, 1:6]REF $P$;
```

Routines Used in Manipulating Compiler List Structures

The storage structures used in Figure 1 are similar to ones already suggested by Goos (9). To construct them, the compiler subroutines use a package of FORTRAN subroutines derived from the primitives of Weizenbaum's SLIP system (10). These SLIP-inspired primitives are listed and explained briefly in Table 1 below.

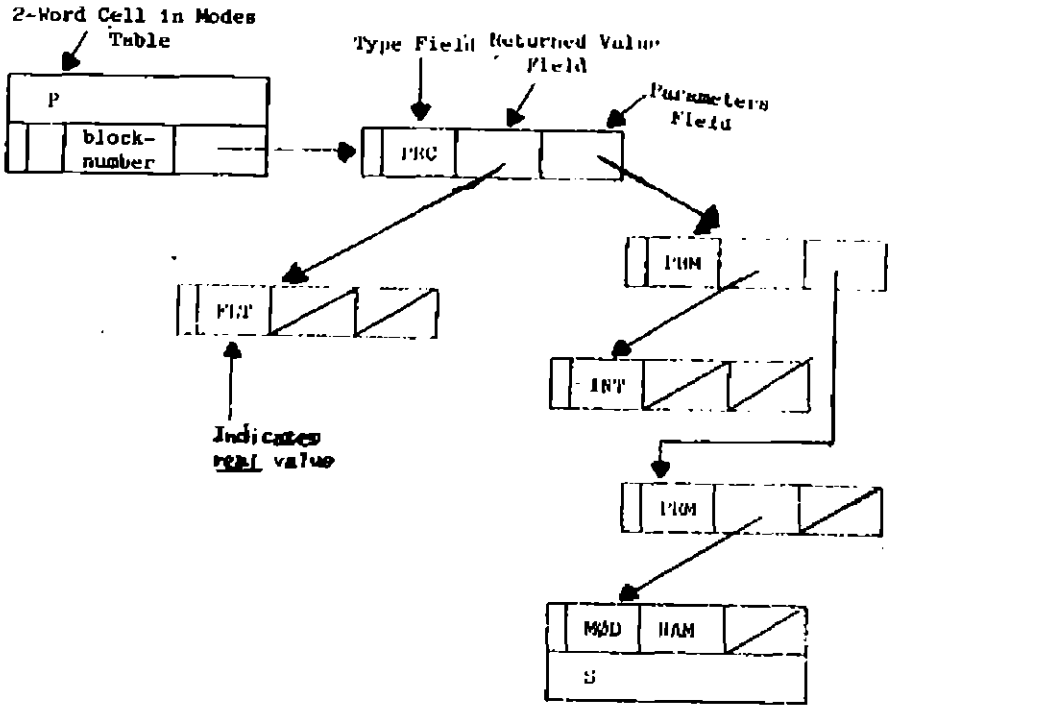
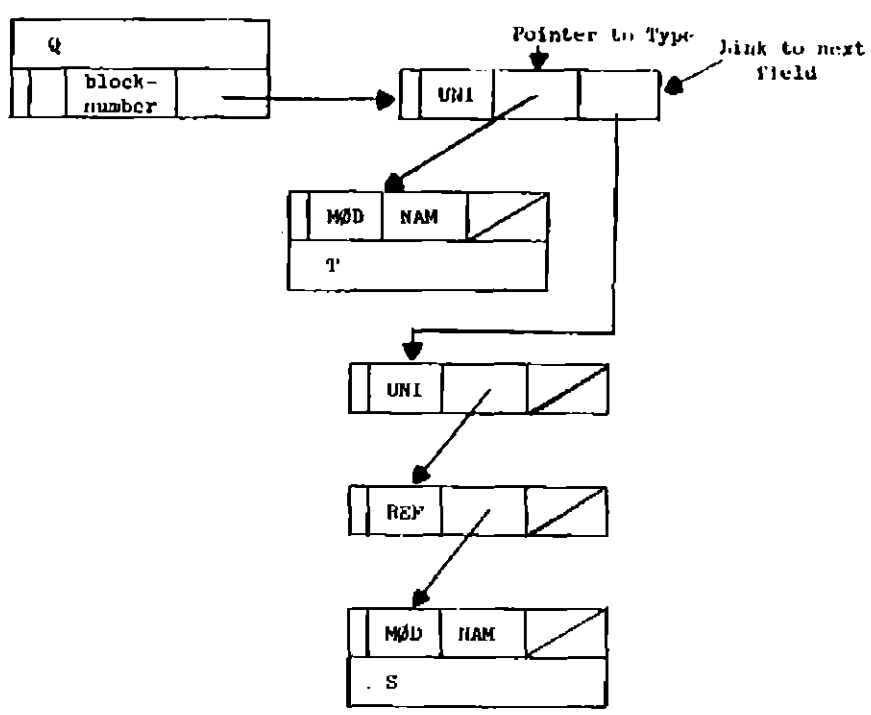


Figure 1. Representation of mode declarations by compiler lists



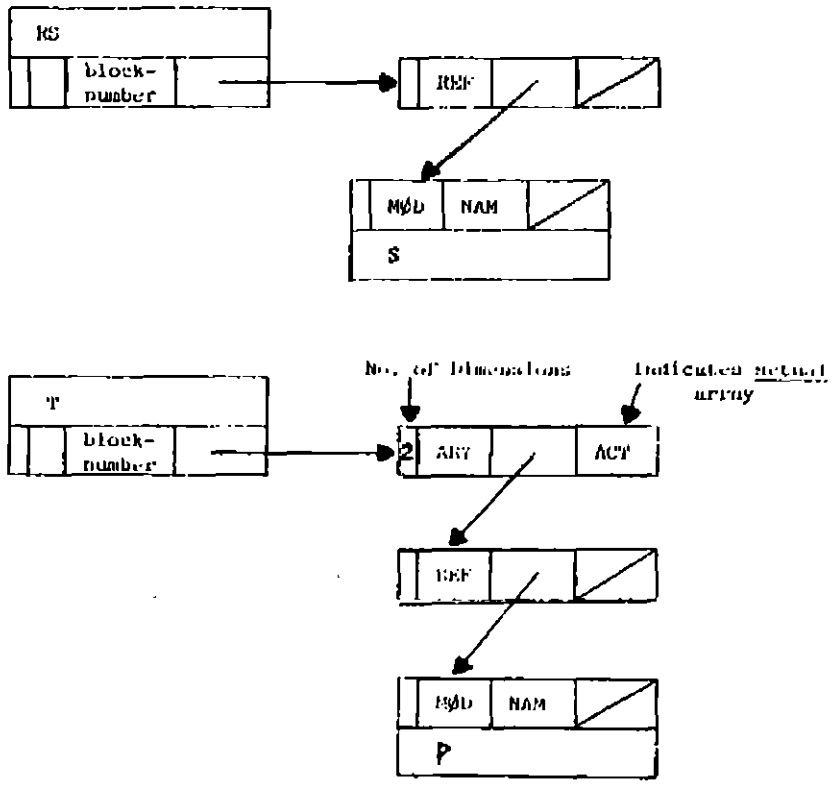


Figure 1. Continued

Table 1. List Processing Primitives Used by the Compiler

Name	Meaning
LØCF(X)	address (X) → LØCF (The function returns the machine address of FØRTRAN variable X.)
CØNT(I)	[I] → CØNT
INHALT(I)	[I] → INHALT (The value stored in the memory cell whose address is given by I is returned.)
IBYTE(J,K)	J(6*(K-1) → 6*K) → IBYTE (The Kth character in word J is returned right justified with padded zeroes.)
LINK(J,K)	J(18*(K-1) → 18*K) → LINK (The Kth address-sized field in word J is returned right justified with padded zeros.)
PUTDIR(A,B,K)	A(0 → 5) → B(6*(K-1) → 6*K) (The first character of A is put into the Kth character position of word B.)
PUTIND(A,J,K)	A(0 → 5) → [J](6*(K-1) → 6*K) (The first character of A is put into the Kth character position of the word whose machine address is stored in J.)
SETDIR(A,B,K)	A(0 → 17) → B(18*(K-1) → 18*K)
SETIND(A,J,K)	A(0 → 17) → [J](18*(K-1) → 18*K)

In addition to the functions in Table 1, there is an integer function called IFETCH(K) and a logical function called SAMETYP(I,J,K).

The purpose of IFETCH is to return the machine address of the first word in a list cell of K words stored sequentially in computer memory. The SAMETYP function is, as the name suggests, a function for comparing two compiler data structures to discover if they are of the same "type". Because of the design of ALGOL 68, two declarations may be of the same

"type" if one has any number of intermixed procedure and reference prefixes followed by a copy of the second declaration. Thus, for example, if parameter I pointed to a declaration, such as "PRØC REF REF PRØC REAL", and J pointed to "PRØC REAL", the SAMETYP function would return the FORTRAN .TRUE. value. In addition, after execution of the function, the K parameter would point to a newly-created list containing the sequence "PRØC REF REF". Thus, the list returned to K indicates how many levels of dereferencing and deproceduring must be applied to a datum of type I in order to yield a datum of type J.

The SAMETYP function also incorporates algorithms for looking up the lists assigned to mode indicants appearing in I or J and for replacing these mode indicants in either structure by pointers to these lists. No attempt is made to find reduced versions of structure declarations. However, when two structures are compared that are structurally equivalent as defined in the ALGOL 68 report, a link to the smaller of the two structures replaces the larger structure in the compiler mode table. This structure comparison and replacement algorithm is essentially the same as the one given by Koster (11) and discussed in Goos (9).

Intermediate Language Generated by the Compiler

Since this report is the beginning of an attempt at producing a formal specification of ALGOL 68 that is an alternative to the Van Wijngaarden and the Vienna notations (12), it was decided that the code produced by our compiler subroutines would be in a systems language rather than in the C.D.C. COMPASS assembly language. The systems language chosen is the PILOT language of M.H. Halstead, versions of which have been implemented on the UNIVAC 1108, the C.D.C. 6000 series of computers, the IBM 360/44, and miscellaneous other machines. PILOT is a

much-distilled subset of the original NELIAC language (13), with some useful features, such as the intermixing of PILOT and machine code where desirable in a program, the use of machine addresses for indirect referencing of variable names, and the addition of partial-word masking operators. In PILOT, commas are used to separate statements, expressions can consist of at most two operands separated by an operator, and assignment of value is to the right, rather than to the left, as in ALGOL 60 or FORTRAN. A typical sequence of PILOT statements might be the following:

```
        0 → INDEX REGISTER 1,
        U - 1 → U,
LABEL 1:  INDEX REGISTER 1 + 1 → INDEX REGISTER 1,
        INDEX REGISTER 1 > SIZE : LABEL 2.;;
        LHCURR + 1 → [U + INDEX REGISTER 1],
        LABEL 1.,
LABEL 2:
```

In the statements above, index register 1 is first initialized to zero, and then a loop is entered at label 1. In the loop, index register 1 is compared to the variable "SIZE". If greater than "SIZE", the statement "LABEL 2." (go to LABEL 2) is executed. Otherwise, execution continues with the value of "LHCURR + 1" stored into the memory location whose address is "U + INDEX REGISTER 1". Further examples of PILOT code are given in Section III.

Preprocessing Implied by Rules of the Grammar

When a context-free grammar is actually used as the input program of a compiler-writing system, it has to be written with greater care than one ordinarily suspects. In the first place, what is to be done with a grammar

having reserved words that resemble variable names? In Appendix 1, we see that reserved words like "REAL", "PRØC", "CASE", etc. could just as well be interpreted as program variables. Next, when we look more closely, we see that the syntax for <name> involves individual letters and digits, whereas reserved words appear as groups of letters. Necessarily, the compiler generated from the grammar above will expect to scan successive words of memory, some of which contain only single letters or digits and some of which contain entire reserved words. Hence, a preprocessor subroutine is needed to read in programs to the compiler.

Along with these trivial operations of packing reserved program words for the compiler, the Appendix 1 syntax calls for the preprocessor to supply a priority digit for each expression operator encountered in a program. Since our grammar assigns unique denotations to operators, the preprocessor need only maintain a table of declared operator priorities in each scanned program block. From this table, a priority digit is inserted following each operator in programs read into the compiler. Naturally, priority declarations are no longer needed by the compiler, and so the preprocessor does not supply them to the compiler.

Stack Mechanisms Used by Compiler Subroutines

As in other compilers for block-structure programming languages, our compiler subroutines use tables to store information about currently valid program variables, labels, data types, and operators. In addition, there are two stacks whose purpose is to facilitate our single-pass translation scheme. The first stack consists of two one-dimensional FORTRAN arrays using the same index, "I". These arrays are referred to as "N(I)" and "TYPE(I)", respectively. As can be seen in the compiler subroutines,

the N-stack is used for constructing the compiler representations of data structures, and the TYPE stack carries auxiliary information concerning whether a given declaration is virtual, actual, or formal, and whether it is stowed or nonstowed. Thus, the combined N-TYPE stack carries context information concerning the syntactic objects recognized by the compiler.

"CØDELØC" is the system name for the second stack, a one-dimensional array with index K. As its name implies, the CØDELØC stack saves information about where nested portions of a program appear in its translation. This information is used selectively by the compiler subroutines for inserting and deleting sections of compiled code in the translated program after the compiler has scanned the corresponding segment of the input program. As will be seen in Section III, it is this use of CØDELØC which enables us to produce reasonably efficient and non-redundant code in a single-pass compilation process. Of course, CØDELØC is also used to supply "target locations" for the jumps implicit in the translation of conditional statements.

Using the N-stack alone, it is easy to demonstrate that a translation grammar specifies a compiler that accepts context-sensitive languages like $\{a^n b^n c^n : n > 0\}$. Next, with the addition of the CØDELØC stack, we see that the resulting compiler system potentially has the same computational ability as a Turing machine. This is because any one of the compiler subroutines could be written so as not to return control to the compiler, and could continue operation by manipulating the two stacks as though they were the tape of a Turing machine. Since the compiler system can be made as general as a Turing machine, any multi-pass compiling algorithm can in theory be written in a single-pass version for a translation grammar. Our goal is to write an efficient and rapid single-pass version of what is usually considered a multi-pass compiler system.

Criteria for Selecting Rule Forms in a Translation Grammar

The criteria for selecting certain rule forms in Appendix 1 rather than others have principally to do with the one-pass nature of the compiler-writing system chosen. For example, a one-pass algorithm cannot efficiently tolerate temporary ambiguities in any subtree of a program. As a consequence, the following rules were chosen that are restrictions of the ALGOL 68 language:

<mode indicant> → $<name>$

<operator> → +| -| ... | ↑ <name> ↑

<call> → <primary> (<actual parameters>)

<slice> → <primary> [<indexers>]

<base> → GØ TØ <label>

In the first two rules above, we have forced mode indicants and invented operators to be different in appearance from each other. The next two rules make it impossible to confuse a subscripted variable with a procedure call, and the last rule assures us that the statement

v: = if a then b else c fi;

does not mean the same thing as

v: = if a then b else go to c fi;

which is currently a valid interpretation in the ALGOL 68 report.

Again, the desire for an uncomplicated single-scan compiler led us to abandon the ALGOL 68 definition of a block (given in Section 6.1.1 of the report) in favor of a restricted subset of the definition. In the full ALGOL 68 version of a block, the declaration prelude terminates at the first labeled statement or the first statement followed by a jump. To translate such a declaration prelude properly, the compiler needs to know at the beginning of each statement in a declaration prelude what

comes after that statement. Such knowledge could be gained by a separate pass or by requiring forward scans before each statement, but this extra work consumes compiler time that could be better spent doing translation. Hence, our version of the ALGOL 68 block (rules P_{34} and P_{35} in Appendix 1) is a subset of the full version in which the declaration prelude terminates at the first executable statement in the block. No generality is lost in this version by not interspersing statements and declarations in the declaration prelude; rather, these interspersed statements are made to appear within the declarations in the blocks that legally may appear there.

The preceding problems of declaration preludes and uniquely identifiable mode and operator indicants are all examples of avoiding syntaxes that result in nondeterministic compilers (6). Another example of this problem is one that arose when an attempt was made to transcribe a portion of the ALGOL 68 report directly into the notation of Appendix 1. In this case the syntax for <stowed declarer> initially took the form:

<stowed declarer> → STRUCT (<fields>)
| [<rowers>] STRUCT (<fields>)
| [<rowers>] <nonstowed declarer>
| [<rowers>] <mode indicant>

With this set of rules, we are left with three separate methods to declare an array and no way of knowing initially which compiler subroutines should be called. Because of this, the following version of the syntax of arrays and structures appears in Appendix 1:

<stowed declarer> → STRUCT (<fields>)

| <array generator> STRUCT (<fields>)

| <array generator> <nonstowed declarer>

| <array generator> <mode indicant>

<array generator> → [<rowers>]

Apart from the necessity of avoiding nondeterminisms in the compiler, the other main criterion for selecting certain configurations of rules in a translation grammar is to permit syntax-directed modifications of compiler tables and syntax-directed tests for consistency of program subtrees with non-syntactic objects. As an example of simple modifications made possible by syntax, the following translation rules cause an extra level of reference to be prefixed on the compile-time representation of the declarer in question:

P_{29a}: <reference to actual declarer> → $<^{\text{actual mode}}_{\text{declarer}}>/\psi_{29a1}/$

P_{2a} : <nonstowed declarer> → REF/ψ_{2a1}/<mode declarer>/ψ_{2a2}/

P_{30a}: <reference to mode global generator> → HEAP/ψ_{30a1}/
<actual mode declarer>/ψ_{30a2}/

In the second example, the rules

P_{21a}: <virtual mode declarer> → <mode declarer>/ψ_{21a}/

P_{22a}: <actual mode declarer> → <mode declarer>/ψ_{22a}/

do not introduce any syntactic ambiguity in the language, since the corresponding transitions of the compiler can only occur in mutually exclusive contexts. However, compiler subroutines ψ_{21a} and ψ_{22a} both test the N-TYPE stack to see that all arrays declared within the <mode declarer> are respectively virtual or actual.

Lastly, we see that syntactic rules may be chosen to make the work of compiler subroutines easier. Good examples of this are the sets of rules below:

P_{5a}: <fields> → <field>/ψ_{5a1}/

P_{5b}: <fields> → <field,> <fields>/ψ_{5b1}/

P_{6a}: <field,> → <field>,/ψ_{6a}/

Subroutine ψ_{6a} generates a compile-time list cell for each field in a structure declaration, and subroutine ψ_{5b1} links together the list cells generated in this fashion. Because of the form of P_{6a}, the structure fields are stored in sequence on the N-TYPE stack by ψ_{6a}. Rule P_{5b} calls for the compiler to work backwards, linking together the two structure fields on top of the N-TYPE stack by successive calls on subroutine ψ_{5b1}. Thus the syntax directs the compiler to link together the compile-time representation of the structure.

On the other hand, the rules

P_{7a}: <rowers> → <rower>/ψ_{7a1}/

P_{7b}: <rowers> → <rowers>, <rower>/ψ_{7b1}

do nothing more than construct a run-time array descriptor in the order of appearance of the program.

III. Translation of Array and Structure Declarations

Preliminaries

ALGOL 68 uses two reservoirs of consecutive memory words for use in generating arrays and structures during program execution. In the first reservoir system, array storage is taken from a stack (called loc in the defining report) whose contents are divided into regions corresponding to currently active program blocks. As the most recent block is abandoned by the program at run time, the storage of its corresponding region in loc is freed for reuse, exactly as in an ALGOL 60 system. As a consequence, structures and arrays stored in loc are only accessible during the lifetime of the block in which they were created. In order to permit a longer lifetime of usage for selected arrays and structures, an alternative reservoir system is available, called the heap in the defining report. The heap system provides that an array or structure is only reclaimed when it can no longer be reached from an active program variable. The use of this heap reservoir necessitates use of a conventional list-processing storage reclamation scheme at run time.

During compilation, the compiler subroutines keep track of whether a currently translated declaration is to be stored in loc or heap. The bookkeeping system used for this purpose is quite simple. The FORTRAN variable "LHCURR" is initialized to contain the characters "LØC", and is temporarily reset to contain the characters "HEAP" whenever compiler subroutines ψ_{30a1} and ψ_{30a2} in translation rule P_{30a} are called. Then, during the translation process, whenever the code for a storage reservation is put out, the contents of LHCURR (denoted in our notation below by "{LHCURR}") is inserted in the translated program.

Array Declarations

There are two parts to the problem of generating arrays in a compiled ALGOL 68 program. The first part involves construction of a run-time descriptor for the array, and the second part involves possible assignment of structures to each element of the resulting array. Translation rules P_{4a} and P_{7a} of Appendix 1 outline the construction of the descriptor, while rules P_{10a} through P_{13e} keep track of whether the descriptor is virtual (has no numerical bounds), actual (has only numerical bounds) or formal (has some mixture of numerical and non-numerical bounds). Since no code is produced for virtual or formal descriptors, compiler subroutine ψ_{4a1} records in the CØDELØC stack where translated code for the descriptor begins, so that ψ_{4a2} can erase any code that may have been produced while scanning the <array generator>.

For the recognized program sequence

[<rowers>]

the compiler produces the following translated code:

SAVEU,

2 + {LHCURR} → U,

{LHCURR} + $\boxed{\begin{matrix}1 + \text{no. of} \\ \text{dimensions}\end{matrix}}$ → {LHCURR},

0 → [U],

Translation of <rowers>,

U - $\boxed{\begin{matrix}1 + \text{no. of} \\ \text{dimensions}\end{matrix}}$ → TEMP1,

0 → [TEMP1],

[U](36 → 53) → SIZE,

SIZE > 0: U → [TEMP1](0 → 17),

{LHCURR} + SIZE → {LHCURR};;

$\boxed{\begin{matrix}\text{no. of} \\ \text{dimensions}\end{matrix}}$ → [TEMP1](36↱53),

PØPU,

The run-time variable U in the above code is used by the compiler subroutines as an index for placing information in the current word of the array descriptor being generated. As can be seen in Figure 2, the descriptor consists of n+1 words, with n the number of dimensions. Thus, the first descriptor word storing bounds is located at address "2 + {LHCURR}" above (see Preliminaries for meaning of {LHCURR}), and that word is set to zero by the "0 → [U]" statement.

The statement that reserves space for the descriptor in {LHCURR} is inserted later by subroutine ψ_{4a2} after the number of dimensions is known by the compiler. Translation of <rowers> involves calculation of the "strides" of the descriptor. Each stride is associated with a pair of bounds and is the coefficient in a run-time storage-mapping function used in subscripting. When the final stride of the descriptor is calculated, it is stored in the next location following the last pair of bounds in memory and coincides with the actual size of the run-time array. Thus, the code

[U](36 → 53) → SIZE,

above extracts this value and uses it above in a conditional statement that places a link to the first array element into the run-time descriptor:

SIZE > 0: U → [TEMP1](0 → 17),{LHCURR} + SIZE → {LHCURR};;

Figure 2 below displays the storage structure used for run-time array descriptors. The "$\ell_i u_i$" notation represents a two-bit code for indicating whether or not the corresponding lower and upper bounds can be changed dynamically. When ℓ_i (or u_i) is zero, the lower (or upper) bound may vary; otherwise it is fixed. Given this information, we can discuss the translation of <rowers>.

Figure 2. Run-Time Representation of Array Descriptors

Type	No. of Dimensions	Block Number	Pointer to First Element of Array
$\ell_1 u_1$	stride 1	upper bound 1	lower bound 1
.	.	.	.
.	.	.	.
.	.	.	.
$\ell_n u_n$	stride n	upper bound n	lower bound n

Rules P_{7a} and P_{7b} of Appendix 1 are given below:

P_{7a} : <rowers> → <rower>/ψ_{7a1}/

P_{7b} : <rowers> → <rowers>, <rower>/ψ_{7b1}/

Subroutine ψ_{7a1} is called after the translation of the first pair of bounds in an array declaration, and subroutine ψ_{7b1} is called for each subsequent pair of bounds in the declaration. Primarily, ψ_{7a1} and ψ_{7b1} work together to count dimensions and calculate the strides of the descriptor. They assume that translation rules P_{13}, P_{12}, P_{11}, and P_{10} have inserted the upper and lower bounds of each dimension into the appropriate fields of the descriptor word. The code put out by ψ_{7b1} gives an idea of this interaction:

```
[U](18 → 35) - [U](0 → 17) → SIZE,
(Subtracts the lower bound from the upper bound.)
SIZE * [U](36 → 53) → SIZE,
(Calculates the value of the next stride.)
U + 1 → U,
0 → [U],
(Zeroes the next word in the descriptor.)
SIZE > 0 :  SIZE → [U](36 → 53);;
(Stores the stride in the next word of the descriptor.)
```

When the array descriptor is completed, there may follow code for initializing each element of the resulting run-time array. This code is written out for rules P_{3b} and P_{3d}:

P_{3b}: <stowed declarer> → <array generator>/ψ_{3b1}/
STRUCT(/ψ_{3b2}/ <fields>)/ψ_{3b3}

P_{3d}: <stowed declarer> → <array generator>/ψ_{3d1}/
<mode indicant>/ψ_{3d2}/

Essentially, subroutines ψ_{3d1} and ψ_{3d2} set up a loop that links each element of the generated array to a fresh copy of the <mode indicant>. This copy of the <mode indicant> is created by treating the <mode indicant> as a function whose value is the address of the next location of the storage reservoir specified by LHCURR. Since translation of P_{3b} works in a very similar fashion, we will give as our illustration the code produced by ψ_{3d1} and ψ_{3d2}:

```
        0 → INDEX REGISTER 2,
        U - 1 → U,
LABEL 2:
        INDEX REGISTER 2 + 1 → INDEX REGISTER 2,
        INDEX REGISTER 2 > SIZE:  LABEL 3.;;
        {LHCURR} + 1 → [U + INDEX REGISTER 2],
        Translation of <mode indicant>,
        (This includes a jump to the definition of the
        <mode indicant> and return.)
        LABEL 2.,
LABEL 3:
```

Note in the above code that "U - 1" denotes the first address in the region of memory reserved for the stored array, whereas {LHCURR} was set equal to the last address of this region by compiler subroutine ψ_{4a2}. Thus, the first address in any data structure created by the <mode indicant> is given by "{LHCURR} + 1", and this address is the value used to initialize the corresponding array elements iteratively.

Structure Declarations

In our implementation of ALGOL 68, the fields of data structure can store plain values or they can be linked to other structures and arrays. During a declaration, then, code linking a structure field to another structure or array may or may not be called for. On a syntactic basis, we say that such linking code is called for in a structure having some field that corresponds to a <stowed declarer> or <mode indicant>. The code is not compiled in the case of a structure field that corresponds to a <nonstowed declarer>. To simplify our discussion of the translation process, we consider only rule P_{3_1} of Appendix 1, since it uses

essentially the same mechanisms as rule P_{3b}:

P_{3a}: <stowed declarer> → STRUCT(/ψ_{3a1}/<fields>)/ψ_{3a2}/

The code generated by compiler subroutines ψ_{3a1} and ψ_{3a2} is as follows:

SAVE U,

{LHCURR} → U,

{LHCURR} + no. of fields → {LHCURR},

Translation of <fields>,

PØP U,

The compiler subroutines used above in translating <fields> also count the number of fields in the structure. When the count is completed, subroutine ψ_{3a2} is called, and ψ_{3a2} inserts the field count into the instruction preceding the translation of <fields>. Thus, the code produced by ψ_{3a1} and ψ_{3a2} reserves space for the translated structure in LØC or HEAP and initializes the index of structure fields to point to the first field in the structure to be translated. The code put out for a typical "stowed" field of the structure is produced by compiler subroutines ψ_{5a1} or ψ_{6a} as follows:

{no. of intervening nonstowed fields} + U → U,

{LHCURR} + 1 → [U],

Translation of the <field>,

In the code above, the index U retains the address of the most recently initialized field in the compiled structure. In order to initialize the next field, U is incremented, and the address of the first free word in LØC or HEAP is inserted in the word whose address is U. Thus, the run-time field is made to point to the first address in the generated structure corresponding to the translated <field>. Of course, no code at all will appear for a "nonstowed" field.

IV. Translation of Mode Declarations

As seen in rule P_{25a}, a mode declaration has the form

MØDE <mode indicant> = <actual mode declarer>.

Sample code produced for such a declaration by compiler subroutines ψ_{25a1}, ψ_{25a2}, and ψ_{25a3} is as follows:

```
LABEL 5.,
LABEL 4:
Translation of <actual mode declarer>,
[RETURN] .,
LABEL 5:
```

In the above code, "LABEL 4" is the unique program label corresponding to the definition of the <mode indicant>. "LABEL 5" is used to isolate the indicant definition in the run-time program. With this isolation, the only way to execute the code produced for <actual mode declarer> is to jump to LABEL 4. After this declarer is executed at run-time, there follows the instruction

```
[RETURN] .,
```

which is used in the PILOT language for a jump to the location whose address is stored in RETURN.

In order to execute this data-definitional function, we need a function call in the form of a <mode indicant> appearing in some declaration. This function call produces the following code:

```
SAVE RETURN,
J + 2 → RETURN,
LABEL 4.,
PØP RETURN
```

For this calling sequence, LABEL 4 is used as the location of the particular <mode indicant> to be called. The PILOT run-time program counter stored in J is saved in the RETURN variable so that the code for the mode definition can jump back to the statement "PØP RETURN". All the compiler needs to do is carry a record of the labels that correspond to the currently active mode indicants and to retain a representation of the data structures to which these indicants correspond. As before, the value of this mode indicant function is the first free location in LØC or HEAP at the calling point in the compiled program.

A Final Example

To bring together the information given in our previous descriptions, we present a typical mode declaration, together with its generated code:

The Declaration

```
MØDE $A$ = STRUCT([1:5] STRUCT($B$ X, REAL Y)Z);
```

The Compiled Code

```
LABEL 10.,
LABEL 7:
SAVE U,
LØC → U,
LØC + 1 → LØC,
U + 1 → U,
LØC → [U],
SAVE U,
2 + LØC → U,
LØC + 2 → LØC,
0 → [U],
1 → [U](35 → 53),
1 → ØPERAND REGISTER 2,
1 → [U](55 → 55),
ØPERAND REGISTER 2 → [U](0 → 17),
5 → ØPERAND REGISTER 2,
1 → [U](54 → 55),
ØPERAND REGISTER 2 → [U](18 → 35),
1 → [U](36 → 53),
[U](18 → 35) - [U](0 → 17) → SIZE,
SIZE + 1 → SIZE,
U + 1 → U,
0 → [U],
SIZE > 0: SIZE → [U](36 → 53);;
```

```
U - 2 → TEMP1,
0 → [TEMP1],
[U](36 → 53) → SIZE,
SIZE > 0: U → [TEMP1](0 → 17),
LØC + SIZE → LØC;;
1 → [TEMP1](36 → 53),
PØP U,
0 → INDEX REGISTER 3,
U-1 → U,
LABEL 8:
INDEX REGISTER 3 + 1 → INDEX REGISTER 3,
INDEX REGISTER 3 > SIZE: LABEL 9.;;
LØC + 1 → [U + INDEX REGISTER 3],
SAVE U,
LØC → U,
LØC + 2 → LØC,
1 + U → U,
LOC + 1 → [U],
SAVE RETURN,
J + 2 → RETURN,
LABEL {$B$}.,
PØP RETURN
PØP U,
LABEL 8.,
LABEL 9:
PØP U,
[RETURN] .,
LABEL 10:
```

Bibliography

References Cited in Text

1. Van Wijngaarden, A. (Ed.) Report on the algorithmic language ALGOL 68. Numerische Mathematik 14 (1969), 79-218.

2. Lindsey, C. H. ALGOL 68 with fewer tears. Algol Bulletin 28 (July 1968), 9-49.

3. Peck, J. E. L. An ALGOL 68 companion. Draft paper, University of British Columbia, Vancouver, 1970.

4. Wirth, N. and Weber, H. EULER: A generalization of ALGOL and its formal definition: parts I and II. Comm. ACM 3 (Jan. - Feb. 1966), 13-25 and 89-99.

5. Lewis, P. M. and Stearns, R. E. Syntax-directed transduction. J. ACM 15 (July 1968), 465-488.

6. Schneider, V. B. A system for designing fast programming language translators. Proc. Spring Joint Comp. Conf. (1969), 777-792.

7. Schneider, V. B. A translation grammar for ALGOL 68. Proc. Spring Joint Comp. Conf. (1970), 493-505.

8. Vere, S. Translation equations. Comm. ACM 13 (Feb. 1970), 83-89.

9. Goos, G. Some problems in compiling ALGOL 68. Technical Report, Rechenzentrum der Technischen Hochschule, Muenchen, Germany, 1970.

10. Weizenbaum, J. A symmetric list processor. Comm. ACM 6 (Sept. 1963), 524.

11. Koster, C. H. A. On infinite modes. ALGOL Bulletin 30 (Feb. 1969), 86-89.

12. Lucas, P. and Walk, K. On the formal description of PL/1. In Annual Review in Automatic Programming 6, 3 (1969), 105-182.

13. Halstead, M. Machine-Independent Computer Programming. Spartan Books, 1962.

Appendix 1

A Partial Translation Grammer for ALGOL 68*

P_{1a}: <mode declarer> → <stowed declarer>/Ψ_{1a}/

P_{1b}: |nonstowed declarer>/Ψ_{1b}/

P_{1c}: |mode indicant>/Ψ_{1c}/

P_{2a}: <nonstowed declarer> → REF/Ψ_{2a1}/ mode declarer /Ψ_{2a2}/

P_{2b}: |mode declarator /Ψ_{2b}/

P_{3a}: <stowed declarer> → STRUCT (/Ψ_{361}/<fields>)/Ψ_{3a2}/

P_{3b}: |<arraygenerator>/Ψ_{361}/STRUCT(/Ψ_{362}/<fields>)/Ψ_{363}/

P_{3c}: |<arraygenerator>*nowstowed declarer>/Ψ_{3c}/

P_{3d}: |<arraygenerator>/Ψ_{3d1}/<mode indicant>/Ψ_{3d2}/

P_{4a}: <arraygenerator> → [/Ψ_{4a1}/ rowers]/Ψ_{4a2}/

P_{5a}: <fields> → <field>/Ψ_{5a1}/

P_{5b}: |<field>,/Ψ_{5b1}/<fields>/Ψ_{5b2}/

P_{6a}: <field> → <field mode declarer><field selector>/Ψ_{6a}/

P_{7a}: <rowers> → <rower>/Ψ_{7a1}/

P_{7b}: |<rowers>, <rower>/Ψ_{7b1}/

P_{8a}: <field mode declarer> → <mode declarer>

P_{9a}: <field selector> → <name>/Ψ_{9a}/

P_{10a}: <rower> → : /Ψ_{10a}/

P_{10b} |<lower bound>:<upper bound>

P_{11a}: <lower bound> → <bound>/Ψ_{11a}/

P_{12a}: <upper bound> →><bound>/Ψ_{12a}/

P_{13a}: <bound> →<formula>/Ψ_{13a}/

P_{13b}: |EITHER/Ψ_{13b}/

P_{13c}: |FLEX/Ψ_{13c}/

* This syntax has been tested, and is certified to be LR(1).

P_{13}: |<formula EITHER/Ψ_{13d}/

P_{13e}: | formula FLEX/Ψ_{13e}/

P_{14a}: <mode decarator> → <procdecl>

P_{14b}: |<union mode decl>

P_{14c}: | REAL/Ψ_{14c}/

P_{14d}: | INT/Ψ_{14d}/

P_{14e}: | BØØL/Ψ_{14f}/

P_{14f}: | CHARΨ_{14f}/

P_{15a}: <procdecl> → PRØC/Ψ_{15a}/

P_{15b}: | PRØC/Ψ_{15b1}/<virtual procplan>/Ψ_{15b2}/

P_{16a}: <virtual procplan> → (<virtual parameters>)/Ψ_{16a}/

P_{16b}: |<virtual mode declarer>/Ψ_{16b}/

P_{16c}: |(<virtual parameters>)<virtual mode declarer>/Ψ_{16c}/

P_{17a}: <virtual parameters> → <virtual mode declarer>/Ψ_{17a}/

P_{17b}: |<virtual mode declarer>,/Ψ_{17b1}/<virtual parameters>/Ψ_{17b2}/

P_{18a}: <union mode decl> → UNIØN (/Ψ_{18a1}/<nonunion indicants>)/Ψ_{18a2}/

P_{19a}: <nonunion indicants> → <mode indicant>/Ψ_{19a}/

P_{19b}: |<mode indicant>, /Ψ_{19b1}/<nonunion indicants>/Ψ_{19b2}/

P_{20a}: <mode indicant> → $ <name> $/$\Psi_{20a}$/

P_{21a}: <virtual mode declarer> → <mode declarer>/Ψ_{21a}/

P_{22a}: <actual mode declarer> → mode declarer/Ψ_{22a}/

P_{23}: <formal mode declarer> → <mode declarer>

P_{24a}: <declaration> → <mode declaration>

P_{24b}: |<operator declaration>

P_{24c}: |<initialization>

P_{24d}: |<priority declaration>

P_{25a}: <mode declaration> → MØDE/Ψ_{25a1}/<mode indicant> =/Ψ_{25a2}/
<actual mode declarer>/Ψ_{25a3}/

P_{26a}: <operator declaration> → ØP/Ψ_{26a1}/<op.> = /Ψ_{25a2}/<routine>/Ψ_{26a3}/

P_{27a}: <initialization> → <reference to actual declarer>/Ψ_{27a}/<name list>

P_{27b}: |<reference to mode global generator>/Ψ_{27b}/<name list>

P_{27c}: |<formal mode declarer> <tag> =/Ψ_{27c1}/<tertiary>/Ψ_{27c2}/

P_{28a}: <name list> → <tag>/Ψ_{28a}/

P_{28b}: |<tag> : = /Ψ_{28b1}/<tertiary>/Ψ_{28b2}/

P_{28c}: |<name list>, <tag>/Ψ_{28c}/

P_{28d}: |<name list>, <tag> : = /Ψ_{28d1}/<tertiary>/Ψ_{28d2}/

P_{29a}: <reference to actual declarer> → <actual mode declarer>/Ψ_{29a}/

P_{30a}: <reference to mode global generator> → HEAP/Ψ_{30a1}/<actual mode declarer>/
Ψ_{30a2}/

P_{31a}: <program> → $ENTRY$/Ψ_{31a1}/<particular program>$EXIT$/Ψ_{31a2}/

P_{32a}: <particular program> → <closed clause>

P_{33a}: <closed clause> → BEGIN/Ψ_{33a1}/<serial clause>END/Ψ_{33a2}/

P_{33b}: |(/Ψ_{33a1}/<serial clause>)/Ψ_{33a2}/

P_{33c}: |(/Ψ_{33a1}/<row of clause>)/Ψ_{33a2}/

P_{34a}: <serial clause> → <declaration prelude sequence;><suite of clause train>

P_{34b}: |<declaration prelude sequence;><label completer><suit of clause train>

P_{34c}: |<label completer><suite of clause train>

P_{34d}: |<suit of clause train>

P_{35a}: <declaration prelude> → <unitary clause;><declaration prelude>

P_{35b}: |<declaration>

P_{36a}: <declaration prelude sequence;> → <declaration prelude sequence>;

P_{36b}: <unitary clause;> → <unitary clause>;

P_{36c}: <unitary clause EXIT> → <unitary clause>EXIT

P_{36d}: <label:> → <label>:

P_{37a}: <suite of clause train> → <unitary clause>·

P_{37b}: |<labelstat>

P_{37c}: |<labelstat><suite of clause train>

P_{37d}: |<unitary clause;><suite of clause train>

P_{38a}: <labelstat> → <unitary clause;><label:>

P_{38b}: |<unitary clause EXIT><label:>

P_{38c}: |<labelstat><label:>

P_{38d}: <label completer> → <label:>

P_{38e}: |<label completer><label:>

P_{38f}: <declaration prelude sequence> → <declaration prelude>/Ψ_{34b2}/

P_{38g}: |<declaration prelude sequence;><declaration prelude>

P_{39a}: <label> → <name>

P_{40a}: <unitary clause> → <tertiary>

P_{40b}: |<confrontation>

P_{40c}: |FØR <tag> FRØM <serial clause> BY <serial clause> TØ <serial clause>
DØ <unitary clause>

P_{40d}: |FRØM <serial clause> BY <serial clause> TØ <serial clause> DØ
<unitary clause>

P_{40e}: |WHILE <serial clause> DØ <unitary clause>

P_{41a}: <confrontation> → <identity relation>

P_{41b}: |<conformity relation>

P_{41c}: |<reference to mode assignation>

P_{41d}: |<mode cast>

P_{42a}: <reference to mode assignation> → <reference to mode tertiary>
: = <unitary clause>

P_{43a}: <reference to mode tertiary> → <tertiary>

P_{44a}: <conformity relation> → <reference to mode tertiary>
<conformity relator><tertiary>

P_{45a}: <conformity relator> → ::

P_{45b}: |::=

P_{46a}: <identity relations> → <reference to mode tertiary>
<identity relator><reference to mode tertiary>

P_{47a}: <identity relator> → :=:

P_{47b}: |:=:

P_{48a}: <mode cast> → <virtual mode declarer>/Ψ_{48a1}/<unitary clause>/Ψ_{48a2}/

P_{49a}: <void cast> → :<unitary clause>/Ψ_{49a}/

P_{50a}: <tertiary> → <formula>

P_{51a}: <formula>→ <formula><op.><unary>

P_{51b}: |<unary>

P_{52a}: <unary> → <op.><unary>

P_{52b}: |<secondary>

P_{53a}: <priority declaration> → PRIØRITY<op>=<nonzero digit>

P_{54a}: <moid cast> → <mode cast>

P_{54b}: |<void cast>

(Rules P_{55} through P_{60} of first version are deleted.)

P_{61a}: <op.> → +

P_{61b}: | -

P_{61c}: | *

P_{61d}: | /

P_{61e}: | /.

P_{61f}: | ÷

P_{61g}: | ↑

P_{61h}: | ¬

P_{61i}: | ≠

P_{61j}: | ≡

P_{61k}: | >

P_{61l}: | <

P_{61m}: | ≥

P_{61n}: | ≤

P_{61o}: | =

P_{61p}: | ≠

P_{61q}: | -:=

P_{61r}: | +:=

P_{61s}: | *:=

P_{61t}: | /:=

P_{61u}: | ./.:=

P_{61v}: | RE

P_{61w}: | IM

P_{61x}: | ØDD

P_{61y}: | SIGN

P_{61z}: | RØUND

P_{61aa}: |↑<name>↑

P_{62a}: <secondary> → <primary>

P_{62b}: |<cohesion>

P_{63a}: <cohesion> → <selection>

P_{63b}: |<generator>

P_{64a}: <selection> → <name> ØF <secondary>

P_{65a}: <generator> → <reference to mode global generator>

P_{65b}: |<reference to mode local generator>

P_{66a}: |<reference to mode local generator> → LOC <actual mode declarer>

P_{67a}: <primary> → <base>

P_{67b}: |<closed clause>

P_{67c}: |<conditional clause>

P_{68a}: <base> → <tag>

P_{68b}: |<denotation>

P_{68c}: |<slice>

P_{68d}: |<call>

P_{68e}: |GØ TØ <label>

P_{68f}: |SKIP

P_{68g}: |NIL

P_{69a}: <tag> → <name>

P_{70a}: <name> → <letter>

P_{70b}: |<name> <letter>

P_{70c}: |<name> <digit>

P_{70d}: |<name>●

P_{71a}: <denotation> → <number>

P_{71b}: |<character denotation>

P_{71c}: |<string denotation>

P_{71d}: |<bits>

P_{71e}: |<routine>

P_{72a}: <routine> → (*(<formal parameters>)<moid case>*)

P_{72b}: |<moid cast pack>

P_{73a}: <moid cast pack> → (*/Ψ_{73a1}/<moid cast>*)/Ψ_{73a2}/

P_{74a}: <formal parameters> → <formal parameters>

P_{74b}: |<formal parameter;> <<formal parameter>;

P_{75a}: <formal parameter;> → <formal parameter>;

P_{76a}: <formal parameter> → <formal mode declarer> <name>

P_{77a}: <call> → <primary>(/Ψ_{77a1}/<actual parameters>)/Ψ_{77a2}/

P_{78a}: <actual parameters> → <unitary clause>/Ψ_{78a1}/

P_{78b}: |<unitary clause>,/Ψ_{78b1}/<actual parameters>/Ψ_{78b2}/

P_{79a}: <slice> → <primary> [<indexer>]

P_{80a}: <indexer> → <trimscript>

P_{80b}: |<indexer>, <trimscript>

P_{81a}: <trimscript> → <sum>

P_{81b}: | : AT <sum>

P_{81c}: |<sum> : <sum> AT <sum>

P_{81d}: |<sum> : AT <sum>

P_{81e}: |<sum> : <sum>

P_{81f}: | : <sum> AT <sum>

P_{82a}: <conditional clause> → IF/Ψ_{33a1}/<serial clause>
THEN/Ψ_{82a2}/<serial clause>
ELSE/Ψ_{82a3}/<serial clause>FI/Ψ_{82a4}/

P_{82b}: |IF/Ψ_{33a1}/<serial clause>THEN/Ψ_{82a2}/<serial clause>FI/Ψ_{82b3}/

P_{82c}: |CASE/Ψ_{33a1}/<serial clause>IN/Ψ_{82d2}/<row of clause>ESAC/Ψ_{82c3}/

P_{82d}: |CASE/Ψ_{33a1}/<serial clause>IN/Ψ_{82d2}/<row of clause>ELSE/Ψ_{82d3}/
<serial clause>ESAC/Ψ_{82d4}/

P_{83a}: <row of clause> → <unitary clause,><unitary clause>/Ψ_{83a}/

P_{83b}: |<unitary clause,><row of clause>/Ψ_{83b}/

P_{84a}: <unitary clause,> → <unitary clause>,/Ψ_{84a}/

P_{85a}: Deleted

P_{86a}: <number> → <a integer>

P_{86b}: |<a integer> . <b integer>

P_{86c}: |<a integer>.

P_{86d}: |.<b integer>

P_{87a}: <a integer> → <nonzero digit>

P_{87b}: |<a integer> <digit>

P_{88a}: <b integer> → <digit>

P_{88b}: |<b integer> <digit>

P_{89a}: <bits> → 0 <octal digit>

P_{90a}: <octal integer> → <octal digit>

P_{90b}: |<octal integer> <octal digit>

P_{91a}: <octal digit> → 0

P_{91b}: |<nonzero octal>

P_{92a}: <nonzero octal> → 1

.

.

.

.

P_{92g}: | 7

P_{93a}: <digit> → 0

P_{93b}: |<nonzero digit>

```
      SUBROUTINE CLEAN                                                  10
      ENTRY PSI1A                                                       20
C                                                                       30
      ENTRY PSI1C                                                       40
C                                                                       50
      TYPE(I)=6RSTOWED                                                  60
      RETURN                                                            70
C                                                                       80
      ENTRY PSI1B                                                       90
C                                                                      100
      TYPE(I)=9RNONSTOWED                                              110
      RETURN                                                           120
C                                                                      130
      ENTRY PSI2A1                                                     140
C                                                                      150
      K=K+1                                                            160
      CODELOC(K)=Z                                                     170
      RETURN                                                           180
C                                                                      190
      ENTRY PSI2A2                                                     200
C                                                                      210
C CODE FOR THIS NONSTOWED DECLARER IS DESTROYED AT THIS POINT. INFORMATI 220
C ABOUT FORMAL DECLARERS IS PASSED ON.                                 230
C                                                                      240
      IF (LINK(CONT(N(I)),1).EQ.3RFRM) GO TO 101                       250
      ITEMP1=3RVCT                                                     260
      GO TO 102                                                        270
  101 ITEMP1=3RFRM                                                     280
  102 ITEMP=IFETCH(1)                                                  290
      CALL STRIND (4LOREF,ITEMP)                                       300
      CALL SETIND (N(I),ITEMP,2)                                       310
      CALL SETIND (ITEMP1,ITEMP,1)                                     320
      N(I)=ITEMP                                                       330
      Z=CODELOC(K)                                                     340
      CALL REDUCE (K)                                                  350
      RETURN                                                           360
C                                                                      370
      ENTRY PSI2B                                                      380
C                                                                      390
      CALL SETINC (3RVCT,N(I),1)                                       400
      RETURN                                                           410
C                                                                      420
      ENTRY PSI3A1                                                     430
C                                                                      440
      ENTRY PSI3B2                                                     450
C                                                                      460
C                                                                      470
C A STORAGE RESERVATION COMMAND FOR THE STRUCTURE IS INSERTED HERE     480
C BY SUBROUTINE PSI3A2                                                 490
C                                                                      500
      K=K+1                                                            510
      CODELOC(K)=Z+3                                                   520
      K=K+1                                                            530
      CODELOC(K)=Z+4                                                   540
      FIELDNO(K)=1                                                     550
```

```
      CALL OUTCDA2 (Z,6HSAVEU,,10H        //)                          560
      CALL OUTCDA2 (Z,LHCURR,10H ↑ U,    //)                           570
C                                                                        580
C CODE FOR THE OUTER STRUCTURE AND ITS FIRST FIELD IS INSERTED HERE BY   590
C RULE PSI3A2.                                                           600
C                                                                        610
      Z=Z+2                                                              620
      CALL OUTCDA2 (Z,LHCURR,10H ↑ [U]  //)                              630
      RETURN                                                             640
C                                                                        650
      ENTRY PSI3A2                                                       660
C                                                                        670
C ALL CODE FOR FIELDS HAS BEEN GENERATED BY THIS POINT. THE REMAINING    680
C CODE FOR THE OUTER STRUCTURE IS INSERTED HERE.                         690
C                                                                        700
      ITEMP=IFETCH(1)                                                    710
      CALL STRTNO (4LOSTR,ITEMP)                                         720
      CALL SETIND (N(I),ITEMP,2)                                         730
      CALL SETIND (LINK(CONT(N(I)+1),1),ITEMP,1)                         740
      COUNTNO=IBYTE(CONT(N(I)),10)                                       750
      CALL PUTIND (COUNTNO,ITEMP,10)                                     760
      N(I)=ITEMP                                                         770
      CALL REDUCE (K)                                                    780
C                                                                        790
C CLEARS THE STACK OF INFORMATION REGARDING THE FIELDS.                  800
C                                                                        810
      INDEX=CODELOC(K)                                                   820
      CALL OUTCDA6 (Z,LHCURR,2H +,CONVERT(COUNTNO,2R  ),2H ↑,LHCURR,7H,  830
     1   //)                                                            840
      IF (COUNTNO.GT.FIELDNO(K)) GO TO 103                               850
C                                                                        860
C IF EQUAL, THIS MEANS THAT NONE OF THE STRUCTURE FIELDS ARE STORED,     870
C AND HENCE, NO SUBSTRUCTURES WERE CREATED.                              880
C                                                                        890
      ITEMP=CODELOC(K)-2                                                 900
      CALL OUTCDB1 (ITEMP,2H//)                                          910
      CALL OUTCDB1 (ITEMP,2H//)                                          920
      GO TO 104                                                          930
  103 CALL OUTCDA1 (Z,10HPOPU,    //)                                   940
  104 CALL REDUCE (K)                                                    950
      RETURN                                                             960
C                                                                        970
      ENTRY PSI3B1                                                       980
C                                                                        990
      ENTRY PSI3D1                                                      1000
C                                                                       1010
      ITEMP=CONVERT(R,2RIR)                                             1020
      R=R+1                                                             1030
      IF (R.GT.MAXR) MAXR=R                                             1040
      CALL OUTCDA3 (Z,3H0↑ ,ITEMP,10H,        //)                      1050
      CALL OUTCDA2 (Z,10HU - 1 ↑ U,,10H         //)                     1060
      S=S+1                                                             1070
      K=K+1                                                             1080
      CODELOC(K)=CONVERT(S+1,2R L)                                      1390
      K=K+1                                                             1100
```

```
      CODELOC(K)=CONVERT(S,2R L)                                          1110
      CALL OUTCOA2 (Z,CODELOC(K),10H?         //)                        1120
      CALL OUTCOA4 (Z,ITEMP,5H + 1?,ITEMP,10H,        //)                  1130
      S=S+1                                                               1140
      CALL OUTCOA4 (Z,ITEMP,9H > SIZE ?,CODELOC(K-1),10H. ??     //)       1150
      CALL OUTCOA4 (Z,LHCURR,10H+1 ? (U + ,ITEMP,10H),       //)          1160
      RETURN                                                              1170
                                                                          1180
      ENTRY PSI383                                                        1190
                                                                          1200
THISROUTINE IS ALMOST PSI3A2 FOLLOWED BY PSI3D2,WITHFACILITIES FOR        1210
CARRYING OVER INFORMATION ABOUT VIRTUAL ORACTUALDECLARATIONS.              1220
                                                                          1230
      ITEMP=IFETCH(1)                                                     1240
      CALL STRINC (4LOSTR,ITEMP)                                          1250
      CALL SETIND (N(I),ITEMP,2)                                          1260
      CALL SETINC (LINK(CONT(N(I)+1),1),ITEMP,1)                          1270
      COUNTNO=IBYTE(CONT(N(I)),10)                                        1280
      CALL PUTIND (COUNTNO,ITEMP,10)                                      1290
      N(I)=ITEMP                                                          1300
      CALL REDUCE (K)                                                     1310
      INDEX=CODELOC(K)                                                    1320
      CALL OUTCOB6 (INDEX,LHCURR,5H     +,CONVERT(COUNTNO,2R   ),5H     ?,L  1330
     1HCURR,10H,        //)                                              1340
      IF (COUNTNO.GT.FIELDNO(K)) GO TO 105                                1350
      INDEX=CODELOC(K)-2                                                  1360
      CALL OUTCOB1 (INDEX,2H//)                                           1370
      CALL OUTCOB1 (INDEX,2H//)                                           1380
      GO TO 106                                                           1390
  105 CALL OUTCOA1 (Z,10HPOPU,    //)                                     1400
  106 CALL REDUCE (K)                                                     1410
                                                                          1420
      CALL OUTCOA2 (Z,CODELOC(K),10H. ,      //)                         1430
      K=K-1                                                               1440
      CALL OUTCOA2 (Z,CODELOC(K),10H?        //)                         1450
      CALL REDUCE (K)                                                     1460
      CALL SETIND (N(I),N(I-1),2)                                         1470
                                                                          1480
AT THIS POINT, INFORMATION ABOUT WHETHER THE STRUCTURE IS VIRTUAL IS       1490
COMBINED WITH THE SAME INFORMATION CONCERNING THE ARRAY GENERATOR.         1500
                                                                          1510
      ITEMP=LINK(CONT(N(I)),1)                                            1520
      ITEMP1=LINK(CONT(N(I-1)),1)                                         1530
      IF (ITEMP.EQ.3RACT.AND.ITEMP1.EQ.3RACT) GO TO 108                   1540
      IF (ITEMP.EQ.3RFRM.OR.ITEMP1.EQ.3RFRM) GO TO 107                    1550
      CALL SETIND (3RVRT,N(I-1),1)                                        1560
      GO TO 108                                                           1570
  107 CALL SETIND (3RFRM,N(I-1),1)                                        1580
  108 I=I-1                                                               1590
      R=R-1                                                               1600
      RETURN                                                              1610
                                                                          1620
      ENTRY PSI3C                                                         1630
                                                                          1640
AT THIS POINT, ARRANGEMENTS ARE MADE FOR SUBSCRIPTING OF BIT AND          1650
```

```
C CHARACTER ARRAYS. THERE IS NO NEED TO ERASE ANY CODE FOR NONSTOWED          1660
C DECLARERS, SINCE RULES PSI2A2, PSI14, PSI18, AND PSI15 DO THIS.             1670
C                                                                             1680
      ITEMP=LINK(CONT(N(I),3))                                                1690
      IF (ITEMP.EQ.3RBIT) GO TO 110                                           1700
      IF (ITEMP.EQ.3RBYT) GO TO 112                                           1710
  109 CALL SETIND (N(I),N(I-1),2)                                             1720
      I=I-1                                                                   1730
      RETURN                                                                  1740
  110 INC=10H/60 *' (U -                                                      1750
  111 R=R+1                                                                   1760
      S=S+1                                                                   1770
      IF (R.GT.MAXR) MAXR=R                                                   1780
      ITEMP=CONVERT(R,2RIR)                                                   1790
      ITEMP1=CONVERT(S,2R L)                                                  1800
      CALL OUTCDA3 (Z,3H0*' ,ITEMP,10H,        //)                           1810
      CALL OUTCDA8 (Z,ITEMP1,6H* (U- ,ITEMP,9H)(36*53) ,INC,ITEMP,10H)(3      1820
     16*53), ,10H         //)                                                 1830
      CALL OUTCDA4 (Z,ITEMP,6H  +1 *',ITEMP,10H,        //)                    1940
      CALL OUTCDA5 (Z,ITEMP,10H<DIMENSION,1H*,ITEMP1,10H.*;;     //)          1850
      R=R-1                                                                   1860
      GO TO 109                                                               1870
  112 INC=10H12*'(U -                                                         1880
      GO TO 111                                                               1890
C                                                                             1900
      ENTRY PSI3C2                                                            1910
C                                                                             1920
      CALL OUTCDA2 (Z,CODELOC(K),10H. ,      //)                             1930
      K=K-1                                                                   1940
      CALL OUTCDA (Z,CODELOC(K),10H*        //)                              1950
      CALL REDUCE (K)                                                         1960
      CALL SETIND (N(I),N(I-1),2)                                             1970
      I=I-1                                                                   1980
      R=R-1                                                                   1990
      RETURN                                                                  2000
C                                                                             2010
      ENTRY PSI4A1                                                            2020
C                                                                             2030
      K=K+1                                                                   2040
      CODELOC(K)=Z                                                            2050
      FIELDNO(K)=0                                                            2060
      CALL OUTCDA1 (Z,10HSAVEU,  //)                                          2070
      CALL OUTCDA2 (Z,LHCURR,10H+ 2 * U //)                                   2080
      Z=Z+1                                                                   2090
      CALL OUTCDA2 (Z,9H00 *' (U),,10H         //)                           2100
C                                                                             2110
C THIS OUTPUT CODE MAKES U THE INDEX OF THE DESCRIPTOR BEING CREATED.         2120
C A SPACE IS LEFT IN THE OUTPUT CODE FOR LATER INSERTION OF A STATEMENT       2130
C OF THE FORM LHCURR+DIMENSION*' LHCURR THAT RESERVES SPACE FOR THE           2140
C ARRAY DESCRIPTOR ON EITHER THE HEAP OR THE LOC STACK. RULE PSI4A2           2150
C MAKES THIS INSERTION                                                        2160
C                                                                             2170
      RETURN                                                                  2200
C                                                                             2210
      ENTRY PSI4A2                                                            2220
```

```
C                                                                       2230
C CAUSES THE TOTAL SIZE OF THE ARRAY TO BE CALCULATED, SEES WHETHER THE 2240
C ARRAY IS VIRTUAL, AND STORES THE DIMENSION IN THE DESCRIPTOR.         2250
C                                                                       2260
      BOUNDNO=LINK(FIELDNO(K),1)                                        2270
      I=I+1                                                             2280
      N(I)=IFETCH(1)                                                    2290
      CALL PUTIND (BOUNDNO,N(I),10)                                     2300
      CALL SETIND (3RARY,N(I),3)                                        2310
      VIRTUAL=LINK(FIELDNO(K),2)                                        2320
      IF (VIRTUAL.EQ.0) GO TO 115                                       2330
      IF (VIRTUAL.EQ.BOUNDNO) GO TO 113                                 2340
      CALL SETIND (3RFRM,N(I),1)                                        2350
C                                                                       2360
C THE ARRAY IS FORMAL.                                                  2370
C                                                                       2380
      GO TO 114                                                         2390
  113 CALL SETIND (3RVRT,N(I),1)                                        2400
C                                                                       2410
C THE ARRAY IS VIRTUAL, AND ITS ASSOCIATED TRANSLATION IS DESTROYED.    2420
C                                                                       2430
  114 Z=CODELOC(K)                                                      2440
      K=K-1                                                             2450
      RETURN                                                            2460
  115 CALL SETIND (3RACT,N(I),1)                                        2470
C                                                                       2480
C FOR AN ACTUAL ARRAY, WE PROCEDE TO PUT OUT CODE FOR COMPLETING        2490
C THE DESCRIPTOR.                                                       2500
C                                                                       2510
      ITEMP=CODELOC(K)+3                                                2520
      ITEMP1=CONVERT(BOUNDNO,2R  )                                      2530
C                                                                       2540
C WE NOW HAVE THE INFORMATION THAT PSI781 STORED THE SIZE OF THE ARRAY  2550
C IN U AT RUN TIME,WHERE U IS CURRENTLY THE ADDRESS OF THE WORD THAT    2560
C FOLLOWS THE DESCRIPTOR IN MEMORY.                                     2570
C                                                                       2580
      CALL OUTCDP6 (ITEMP,LHCURR,4H   +,ITEMP1,4H   *,LHCURR,5H,  //)   2590
      CALL REDUCE (K)                                                   2600
      CALL OUTCDA4 (Z,3HU -,ITEMP1,8H * TEMP1,10H,       //)            2610
      CALL OUTCDA2 (Z,10H00*(TEMP1),10H,       //)                     2620
      CALL OUTCDA2 (Z,10H(U)(36*53),10H*SIZE,  //)                     2630
      CALL OUTCDA3 (Z,7HSIZE>0*,10HU*(TEMP1)(,10H0*17),  //)            2640
      CALL OUTCDA4 (Z,LHCURR,10H + SIZE * ,LHCURR,10H;;      //)        2650
      CALL OUTCDA3 (Z,ITEMP1,10H-1*(TEMP1),10H(36*53),//)               2660
C                                                                       2670
C AT THIS POINT, U IS THE ADDRESS OF THE FIRST ARRAY ELEMENT.           2680
C                                                                       2690
      RETURN                                                            2700
C                                                                       2710
      ENTRY PSI5A1                                                      2720
C                                                                       2730
      ENTRY PSI6A                                                       2740
C                                                                       2750
C CHECKS THE STRUCTURE FIELD TO DISCOVER WHETHER STOWED OR NONSTOWED. IF 2760
C NONSTOWED, DELETES PREVIOUSLY GENERATED CODE FOR POINTING FROM FIELD  2770
```

```
; TO AVAILABLE STORAGE AT RUN TIME, AND INCREMENTS A GENERAL COUNT OF      2780
; FIELDS. IF STOWED OR MODE INDICANT, PUTS OUT CODE FOR INCREMENTING RUN   2790
; TIME ALLOCATION POINTER TO GENERATE A CONNECTION BETWEEN A FIELD AND     2800
; ITS DATA STRUCTURE.                                                      2810
;                                                                          2820
      IF (TYPE(I).EQ.6RSTOWED) GO TO 116                                   2830
      Z=CODELOC(K)-1                                                       2840
      FIELDNO(K)=1+FIELDNO(K)                                              2850
      GO TO 117                                                            2860
  116 ITEMP=CODELOC(K)                                                     2870
      CALL OUTCDB2 (ITEMP,CONVERT(FIELDNO(K),2R  ),10H+U*U,    //)         2880
      CALL OUTCDB2 (ITEMP,LHCURR,10H*[U],    //)                          2890
      FIELDNO(K)=1                                                         2900
  117 CODELOC(K)=Z+1                                                       2910
      Z=Z+2                                                                2920
      RETURN                                                               2930
;                                                                          2940
      ENTRY PSI5B1                                                         2950
;                                                                          2960
; LINKS TOGETHER TWO FIELDS OF A STRUCTURE AT COMPILE TIME. COUNTS THE     2970
; NUMBER OF FIELDS. COMBINES INFORMATION ABOUT ACTUAL AND VIRTUAL FIELDS   2980
;                                                                          2990
      CALL SETIND (N(I),N(I-1),2)                                          3000
      CALL PUTIND (1+IBYTE(CONT(N(I)),10),N(I-1),10)                       3010
      ITEMP=LINK(CONT(1+N(I)),1)                                           3020
      ITEMP1=LINK(CONT(N(I-1)),1)                                          3030
      IF (ITEMP.EQ.3RACT.AND.ITEMP1.EQ.3RACT) GO TO 120                    3040
      IF (ITEMP.EQ.3RFRM.OR.ITEMP1.EQ.3RFRM) GO TO 118                     3050
      IF (ITEMP.EQ.3RVCT.OR.ITEMP1.EQ.3RVCT) GO TO 119                     3060
      CALL SETIND (3RVRT,N(I-1),1)                                         3070
      GO TO 120                                                            3080
  118 CALL SETIND (3RFRM,N(I-1),1)                                         3090
      GO TO 120                                                            3100
  119 CALL SETIND (3RVCT,N(I-1),1)                                         3110
  120 I=I-1                                                                3120
      RETURN                                                               3130
C                                                                          3140
      ENTRY PSI7A1                                                         3150
C                                                                          3160
      CALL SETDIR (1,FIELDNO(K),1)                                         3170
      CALL OUTCDA2 (Z,10HO1*[U](36*,10H53),    //)                        3180
      CALL OUTCDA4 (Z,10H[U](18*35),10H-[U](0*17),6H*SIZE,,10HU+1*U,  //    3190
     1)                                                                    3200
      CALL OUTCDA4 (Z,5H00*U,,7HSIZE>0:,10HSIZE*[U](3,10H6*53)); //)       3210
      RETURN                                                               3220
C                                                                          3230
      ENTRY PSI7B1                                                         3240
C                                                                          3250
C COUNTS UP THE NUMBER OF DIMENSIONS, STORES THE STRIDE IN THE CURRENT     3260
C DESCRIPTOR WORD, AND ZEROES THE NEXT DESCRIPTOR WORD.                    3270
C                                                                          3280
      CALL SETDIR (1+LINK(FIELDNO(K),1),FIELDNO(K),1)                      3290
      CALL OUTCDA3 (Z,10H[U](18*35),10H-[U](0*17),10H*SIZE,  //)           3300
C                                                                          3310
C CALCULATES UPPERBOUND-LOWERBOUND.                                        3320
```

```
                                                                        3330
      CALL OUTCDA3 (Z,10HSIZE*(U)(3,10H6*53)*SIZE,10H,       //)       3340
      CALL OUTCDA1 (Z,10HU+1*U,  //)                                    3350
      Z=Z+1                                                             3360
      CALL OUTCDA4 (Z,6H0*(U),,7HSIZE>0:,10HSIZE*(U)(3,10H6*53)!! //)   3370
      RETURN                                                            3380
                                                                        3390
      ENTRY PSI8A                                                       3400
                                                                        3410
C GENERATES A CELL IN THE COMPILE-TIME LIST COPY OF A STRUCTURE         3420
C DECLARATION.                                                          3430
                                                                        3440
      ITEMP=IFETCH(2)                                                   3450
      CALL STRIND (N(I),ITEMP+1)                                        3460
      I=I-1                                                             3470
      IF (TYPE(I).EQ.6RSTOWED) GO TO 121                                3480
      CALL STRIND (4L0*FL,ITEMP)                                        3490
      GO TO 122                                                         3500
  121 CALL STRIND (4L1*FL,ITEMP)                                        3510
  122 CALL SETIND (N(I),ITEMP,2)                                        3520
      N(I)=ITEMP                                                        3530
      RETURN                                                            3540
                                                                        3550
      ENTRY PSI9A                                                       3560
                                                                        3570
      I=I+1                                                             3580
      N(I)=NAME                                                         3590
      RETURN                                                            3600
                                                                        3610
      ENTRY PSI10A                                                      3620
                                                                        3630
C THIS ROUTINE COUNTS VIRTUAL BOUNDS IN AN ARRAY DECLARATION.           3640
C                                                                       3650
      CALL SETDIR (1+LINK(FIELDNO(K),2),FIELDNO(K),2)                   3660
C                                                                       3670
C NOTE THAT ANYTHING BUT ZERO IN THE SECOND FIELD OF FIELDNO INDICATES  3680
C THAT THE ARRAY IS NOT AN ACTUAL ARRAY.                                3690
C                                                                       3700
      RETURN                                                            3710
C                                                                       3720
      ENTRY PSI11A                                                      3730
C                                                                       3740
      ITEMP1=10H(U)(55*55)                                              3750
      ITEMP2=10H(U)(0* 17)                                              3760
      GO TO 123                                                         3770
C                                                                       3780
      ENTRY PSI12A                                                      3790
C                                                                       3800
      ITEMP1=10H(U)(54*54)                                              3810
      ITEMP2=10H(U)(18*35)                                              3820
  123 GO TO (124,126,126,125,125), BOUND                                3830
  124 CALL OUTCDA3 (Z,5H 01* ,ITEMP1,10H,       //)                    3840
  125 CALL OUTCDA4 (Z,CONVERT(P,2R T),1H*,ITEMP2,10H,       //)         3850
      P=P-1                                                             3860
      I=I-1                                                             3870
```

```
126 RETURN                                                              3880
                                                                        3890
    ENTRY PSI13B                                                        3900
                                                                        3910
    BOUND=2                                                             3920
    RETURN                                                              3930
                                                                        3940
    ENTRY PSI13C                                                        3950
                                                                        3960
    BOUND=3                                                             3970
    RETURN                                                              3980
                                                                        3990
    ENTRY PSI13A                                                        4000
                                                                        4010
    BOUND=1                                                             4020
    GO TO 127                                                           4030
                                                                        4040
    ENTRY PSI13D                                                        4050
                                                                        4060
    BOUND=4                                                             4070
    GO TO 127                                                           4080
                                                                        4090
    ENTRY PSI13E                                                        4100
                                                                        4110
    BOUND=5                                                             4120
127 IF (SAMETYP(N(I),INTEGER,REF)) GO TO 128                            4130
     1    CALL ERROR(110HNON-INTEGER ARRAY BOUND IN BOUNDSLIST.
                                                                        )
    RETURN                                                              4150
128 CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                               4160
    RETURN                                                              4170
                                                                        4180
    ENTRY PSI14C                                                        4190
                                                                        4200
    ITEMP=3RREA                                                         4210
    GO TO 130                                                           4220
                                                                        4230
    ENTRY PSI14D                                                        4240
                                                                        4250
    ITEMP=3RINT                                                         4260
    GO TO 130                                                           4270
                                                                        4280
    ENTRY PSI14E                                                        4290
                                                                        4300
    ITEMP=3RBIT                                                         4310
    GO TO 130                                                           4320
                                                                        4330
    ENTRY PSI14F                                                        4340
                                                                        4350
    ITEMP=3RBYT                                                         4360
    GO TO 130                                                           4370
                                                                        4380
    ENTRY PSI15A                                                        4390
                                                                        4400
129 ITEMP=3RPRC                                                         4410
```

```
130 I=I+1                                                                4420
    ITEMP2=IFETCH(1)                                                     4430
    CALL SETIND (ITEMP,ITEMP2,3)                                         4440
    RETURN                                                               4450
                                                                         4460
    ENTRY PSI15B1                                                        4470
                                                                         4480
    K=K+1                                                                4490
    CODELOC(K)=Z                                                         4500
    GO TO 129                                                            4510
                                                                         4520
    ENTRY PSI15B2                                                        4530
                                                                         4540
CODE GENERATED DURING ANALYSIS OF THE <VIRTUAL PROCPLAN> IS DESTROYED.   4550
                                                                         4560
    Z=CODELOC(K)                                                         4570
    CALL REDUCE (K)                                                      4580
    RETURN                                                               4590
                                                                         4600
    ENTRY PSI16A                                                         4610
                                                                         4620
    CALL PUTIND (1+IBYTE(CONT(N(I)),10),N(I),10)                         4630
    ITEMP1=2                                                             4640
    GO TO 131                                                            4650
                                                                         4660
    ENTRY PSI16B                                                         4670
                                                                         4680
    ITEMP1=1                                                             4690
131 ITEMP=IFETCH(1)                                                      4700
    CALL SETIND (N(I),ITEMP,ITEMP1)                                      4710
    I=I-1                                                                4720
    CALL SETIND (ITEMP,N(I),2)                                           4730
    RETURN                                                               4740
                                                                         4750
    ENTRY PSI16C                                                         4760
                                                                         4770
    ITEMP=IFETCH(1)                                                      4780
    CALL SETIND (N(I),ITEMP,1)                                           4790
    CALL SETIND (N(I-1),ITEMP,2)                                         4800
    CALL PUTIND (IBYTE(CONT(N(I-1)),10),N(I),10)                         4810
    I=I-2                                                                4820
    N(I)=ITEMP                                                           4830
    RETURN                                                               4840
                                                                         4850
    ENTRY PSI17A                                                         4860
                                                                         4870
    ITEMP1=1                                                             4880
    GO TO 132                                                            4890
                                                                         4900
    ENTRY PSI17B1                                                        4910
                                                                         4920
    ITEMP1=0                                                             4930
132 ITEMP=IFETCH(1)                                                      4940
    CALL SETIND (3RPRM,ITEMP,3)                                          4950
    CALL SETIND (N(I),ITEMP,2)                                           4960
```

```
    CALL PUTIND (ITEMP1,ITEMP,10)                                          4970
    N(I)=ITEMP                                                             4980
    RETURN                                                                 4990
                                                                           5000
    ENTRY PSI1782                                                          5010
                                                                           5020
THIS ROUTINE LINKS TOGETHER THE PARAMETERS IN THE COMPILER COPY OF THE     5030
PARAMETER LIST,AND COUNTS THE NUMBER OF PARAMETERS.                        5040
                                                                           5050
    CALL PUTIND (1+IBYTE(CONT(N(I)),10),N(I-1),10)                         5060
    CALL SETIND (N(I),N(I-1),1)                                            5070
    I=I-1                                                                  5080
    RETURN                                                                 5090
                                                                           5100
    ENTRY PSI18A1                                                          5110
                                                                           5120
    K=K+1                                                                  5130
    CODELOC(K)=Z                                                           5140
    RETURN                                                                 5150
                                                                           5160
    ENTRY PSI18A2                                                          5170
                                                                           5180
    Z=CODELOC(K)                                                           5190
    CALL REDUCE (K)                                                        5200
    RETURN                                                                 5210
                                                                           5220
    ENTRY PSI19A                                                           5230
                                                                           5240
    ENTRY PSI19B1                                                          5250
                                                                           5260
    ITEMP=IFETCH(1)                                                        5270
    CALL SETIND (3RUNI,ITEMP,3)                                            5280
    CALL SETIND (N(I),ITEMP,2)                                             5290
    N(I)=ITEMP                                                             5300
    RETURN                                                                 5310
                                                                           5320
    ENTRY PSI1982                                                          5330
                                                                           5340
    CALL SETIND (N(I),N(I-1),1)                                            5350
    I=I-1                                                                  5360
    RETURN                                                                 5370
                                                                           5380
    ENTRY PSI20A                                                           5390
                                                                           5400
    I=I+1                                                                  5410
    ITEMP=IFETCH(2)                                                        5420
    N(I)=ITEMP                                                             5430
    CALL STRIND (NAME,ITEMP+1)                                             5440
    CALL SETIND (3RMOD,ITEMP,3)                                            5450
    NAME=ALIAS(NAME)                                                       5460
                                                                           5470
C THE ALIAS FUNCTION LOOKS UP THE RUN-TIME PROGRAM LABEL ASSIGNED TO THE   5480
C MODE INDICANT,AND ADDS THE INDICANT TO THE COMPILER DECLARATION TABLE    5490
C IF NECESSARY.                                                            5500
C                                                                          5510
```

```
          CALL OUTCOA4(Z,10HSAVE RETUR,2HN,,10HJ+2+RETURN,3H,//)
        CALL OUTCOA4 (Z,NAME,2H.,,10HPOP RETURN,10H,        //)                5530
        RETURN                                                                5540
C                                                                             5550
        ENTRY PSI21A                                                          5560
C                                                                             5570
C A CHECK IS MADE HERE TO SEE WHETHER THE DECLARER IN QUESTION CAN BE         5580
C USED AS A VIRTUAL DECLARER. NO CHECK IS NEEDED FOR FORMAL DECLARERS.        5590
C                                                                             5600
        ITEMP=LINK(CONT(N(I)),1)                                              5610
        IF (ITEMP.EQ.3RVRT.OR.ITEMP.EQ.3RVCT) RETURN                          5620
            CALL ERROR(110HAN ATTEMPT WAS MADE TO USE A FORMAL OR ACTUAL D
     1ECLARER IN PLACE OF A VIRTUAL DECLARER.                                 )
        RETURN                                                                5640
C                                                                             5650
        ENTRY PSI22A                                                          5660
C                                                                             5670
C A CHECK IS MADE TO SEE WHETHER THE DECLARER IN QUESTION CAN BE USED AS      5680
C AN ACTUAL DECLARER. NO CHECK IS NEEDED FOR FORMAL DECLARERS.                5690
C                                                                             5700
        ITEMP=LINK(CONT(N(I),1))                                              5710
        IF (ITEMP.EQ.3RACT.OR.ITEMP.EQ.3RVCT) RETURN                          5720
            CALL ERROR(110HAN ATTEMPT WAS MADE TO USE A FORMAL OR VIRTUAL
     1DECLARER IN PLACE OF AN ACTUAL DECLARER.                                )
        RETURN                                                                5740
C                                                                             5750
        ENTRY PSI25A1                                                         5760
C                                                                             5770
        ENTRY PSI26A1                                                         5780
C                                                                             5790
        K=K+1                                                                 5800
        CODELOC(K)=Z                                                          5810
C                                                                             5820
C THIS RULE ALLOWS DESTRUCTION OF CODE PRODUCED BY RULE PSI20A.               5830
C                                                                             5840
        RETURN                                                                5850
C                                                                             5860
        ENTRY PSI25A2                                                         5870
C                                                                             5880
        Z=CODELOC(K)+2                                                        5890
C                                                                             5900
C A SPACE IS RESERVED FOR A TRANSFER OF CONTROL AROUND THE MODE               5910
C DECLARATION IN THE TRANSLATED PROGRAM.                                      5920
C                                                                             5930
        CALL OUTCOB2 (Z,NAME,10H*        //)                                5940
C                                                                             5950
C THE MODE INDICANT HAS DECLARED BY PSI20A, AND NAME CONTAINS THE RUN-        5960
C TIME LABEL ASSOCIATED WITH THE INDICANT.                                    5970
C                                                                             5980
        FIELDNO(K)=MODE                                                       5990
C                                                                             6000
C MODE IS THE INDEX OF THE COMPILER MODES TABLE.                              6010
C                                                                             6020
        RETURN                                                                6030
C                                                                             6040
```

```
      ENTRY PSI25A3                                                      6050
;                                                                        6060
      CALL OUTCOA3 (Z,9HRETURN*RJ,8H(30*47),,10HRJ.,    //)             6070
      S=S+1                                                              6080
      ITEMP1=CONVERT(S,2R L)                                             6090
      ITEMP=CODELOC(K)+1                                                 6100
      CALL OUTCOB2 (ITEMP,ITEMP1,10H.,      //)                         6110
      CALL OUTCOA2 (Z,ITEMP1,10H:       //)                             6120
      ITEMP=FIELDNO(K)                                                   6130
      MODES(ITEMP,3)=N(I)                                                6140
;                                                                        6150
; ASSIGNS COMPILER DATA STRUCTURE TO MODE INDICANT.                      6160
;                                                                        6170
      I=I-1                                                              6200
      CALL REDUCE (K)                                                    6210
      RETURN                                                             6220
;                                                                        6230
      ENTRY PSI26A2                                                      6240
;                                                                        6250
; AT THIS POINT, THE OPERATOR HAS BEEN ENTERED INTO THE OPS TABLE. THE   6260
; CURRENT INDEX OF OPS IS OP,AND THE RUN-TIME ALIAS OF THE OPERATOR IS   6270
; STORED IN THE COMPILER VARIABLE NAME, EXACTLY AS FOR THE CASE OF MODE  6280
; INDICANTS.                                                             6290
;                                                                        6300
      Z=CODELOC(K)+2                                                     6310
;                                                                        6320
; A SPACE IS RESERVED FOR A TRANSFER OF CONTROL AROUND THE OPERATOR      6330
; DEFINITION IN THE TRANSLATED PROGRAM.                                  6340
;                                                                        6350
      CALL OUTCOB2 (Z,NAME,10H:       //)                               6360
      FIELDNO(K)=OP                                                      6370
      RETURN                                                             6380
;                                                                        6390
      ENTRY PSI26A3                                                      6400
;                                                                        6410
      CALL OUTCOA3 (Z,9HRETURN*RJ,8H(30*47),,10HRJ.,    //)             6420
      S=S+1                                                              6430
      ITEMP1=CONVERT(S,2R L)                                             6440
      ITEMP=CODELOC(K)+1                                                 6450
      CALL OUTCOB2 (ITEMP,ITEMP1,10H.,      //)                         6460
      ITEMP=FIELDNO(K)                                                   6470
      OPS(ITEMP,3)=N(I)                                                  6480
      CALL OUTCOA2 (Z,ITEMP1,10H:       //)                             6490
      I=I-1                                                              6500
      CALL REDUCE (K)                                                    6510
      RETURN                                                             6520
C                                                                        6530
      ENTRY PSI27A                                                       6540
C                                                                        6550
C THIS ROUTINE ADDS AN EXTRA LEVEL OF REFERENCING TO THE DECLARER,BOTH I 6560
C COMPILER REPRESENTATION AND THE RUN-TIME CODE9  LHCURR CARRIES INFORMA  6570
C ABOUT WHETHER LOC OR HEAP IS IN USE.                                   6580
C                                                                        6590
      S=S+1                                                              6600
      ITEMP=CONVERT(S,2R L)                                              6610
```

```
      FIELDNO(K)=ITEMP                                                    6620
      ITEMP1=CODELOC(K)+1                                                 6630
      CALL OUTCOD6 (ITEMP1,ITEMP,4H' 1+,LHCURR,14H',LHCURR,8H,     //)    6640
      CALL OUTCOD4 (ITEMP1,LHCURR,5H+1' [,LHCURR,10H),      //)           6650
      ITEMP1=ITEMP1-2                                                     6660
      S=S+1                                                               6670
      ITEMP=CONVERT(S,2R L)                                               6680
  133 CALL OUTCOD2 (ITEMP1,ITEMP,10H.,      //)                          6690
      CALL OUTCOD3 (Z,10HRETURN' RJ,10H(30'47),  ,10HRJ.,     //)         6700
      CALL OUTCOD2 (Z,ITEMP,10H:       //)                               6710
      ITEMP=IFETCH(1)                                                     6720
      CALL SETIND (3RREF,ITEMP,3)                                         6730
      CALL SETIND (N(I),ITEMP,2)                                          6740
      N(I)=ITEMP                                                          6750
      RETURN                                                              6760
                                                                          6770
      ENTRY PSI27B                                                        6780
                                                                          6790
; LIKE PSI27A,EXCEPT THAT THE EXTRA LEVEL OF RUN-TIME INDIRECTNESS IS     6800
; SUPPLIED BY PSI30A1.                                                    6810
;                                                                         6820
      S=S+1                                                               6830
      ITEMP=CONVERT(S,2R L)                                               6840
      FIELDNO(K)=ITEMP                                                    6850
      ITEMP1=CODELOC(K)+1                                                 6860
      CALL OUTCOD2 (ITEMP1,ITEMP,10H:       //)                          6870
      S=S+1                                                               6880
      ITEMP1=ITEMP1-1                                                     6890
      ITEMP=CONVERT(S,2R L)                                               6900
      GO TO 133                                                           6910
;                                                                         6920
;                                                                         6950
      ENTRY PSI27C1                                                       6960
;                                                                         6970
; THE TAG RULE PSI68A HAS THE EFFECT OF ENTERING THE NAME ONTO THE NAMES  6980
; TABLE, AND LEAVES THE RUN-TIME ALIAS OF <TAG> IN THE VARIABLE NAME.     6990
; CODE FOR THE VIRTUAL DECLARER IS DESTROYED                              7000
;                                                                         7010
      OPERAND(I)=NAME                                                     7020
          TYPE(I)=9RNONSTORED
      Z=CODELOC(K)                                                        7040
      RETURN                                                              7050
;                                                                         7060
      ENTRY PSI27C2                                                       7070
;                                                                         7080
      IF (SAMETYP(N(I-1),N(I),REF).AND.REF.EQ.0) GO TO 134                7090
          CALL ERROR(110HATTEMPT TO INITIALIZE A DECLARED VARIABLE
     1USING A <TERTIARY> OF A DIFFERENT MODE.                        )
      RETURN                                                              7110
  134     IF(TYPE(I).NE.9RNONSTORED)GO TO 135
      P=P+1                                                               7130
      ITEMP=CONVERT(P,2R T)                                               7140
          CALL OUTCOD4(Z,OPERAND(I),1H',ITEMP,10H,       //)
      OPERAND(I)=ITEMP                                                    7200
  135     I=I-1
```

```
      OPERAND(I)=IHASHB(OPERAND(I),N(I))
C                                                                       7290
C SETS THE TYPE OF THE NAME AND THE REGION WHERE ITS VALUE IS STORED.   7300
C                                                                       7310
      CALL OUTCOA4(Z,OPERAND(I+1),1H*,OPERAND(I),10H,       //)
      RETURN                                                            7330
C                                                                       7340
C                                                                      14740
      ENTRY PSI28A                                                     14370
      ENTRY PSI28C                                                     14380
C                                                                      14390
C ENTER THE<TAG> AND ITS MODE ONTO THE NAME TABLE.                     14400
C                                                                      14410
      NAME=IHASHB(NAME,N(I))                                           14420
      CALL OUTCOA4 (Z,LHCURR,4H+1*[,NAME,10H]:      //)                 14430
      CALL OUTCOA3 (Z,10HSAVE RETUR,10HN,J+2*RETU,10HRN,      //)       14440
      CALL OUTCOA2 (Z,FIELDNO(K),10H.,      //)                         14450
      CALL OUTCOA2 (Z,10HPOP RETURN,10H,       //)                     14460
C                                                                      14470
      ENTRY PSI28B1                                                    14480
      ENTRY PSI28D1                                                    14490
C                                                                      14500
      OPERAND(I)=IHASHB(NAME,N(I))                                     14510
      TYPE(I)=9RNONSTORED                                              14520
      RETURN                                                           14530
C                                                                      14540
      ENTRY PSI28B2                                                    14550
      ENTRY PSI28D2                                                    14560
C                                                                      14570
C THE DATA TYPES OF <TERTIARY> AND <TAG> ARE COMPARED. THE <TERTIARY> I 14580
C DEREFERENCED  AND DEPROCEDURED TO ONE LEVEL OF REFERENCE BELOW N(I-1) 14590
C NOTE THAT THE FIRST PARAMETER OF SAMETYP IS THE ONE TO BE LOWERED.    14600
C                                                                      14610
      IF (SAMETYP(N(I),LINK(CONT(N(I-1)),1),REF)) GO TO 169            14620
      CALL ERROR (110HTHE <TERTIARY> ASSIGNED TO THE <TAG> OF THIS MODE 14630
     1DECLARATION DOES NOT POSSESS THE SAME MODE AS ITS ASSIGNEE. )    14640
      RETURN                                                           14650
  169 IF (TYPE(I).NE.9RNONSTORED) GO TO 170                            14660
      P=P+1                                                            14670
      ITEMP=CONVERT(P,2R T)                                            14680
      CALL OUTCOA4 (Z,OPERAND(I),1H*,ITEMP,10H,       //)              14690
  170 CALL OUTEVAL (P,REF,OUTCODE,7,1000-Z)                            14700
      I=I-1                                                            14710
      CALL OUTCOA4 (Z,ITEMP,1H*,OPERAND(I),10H,       //)              14720
      RETURN                                                           14730
C                                                                       7740
      ENTRY PSI29A1                                                     7750
C                                                                       7760
      ITEMP=IFETCH(1)                                                   7770
      CALL SETIND (3RREF,ITEMP,3)                                       7780
      CALL SETIND (N(I),ITEMP,2)                                        7790
      N(I)=ITEMP                                                        7800
      RETURN                                                            7810
C                                                                       7820
      ENTRY PSI30A1                                                     7830
```

```
                                                                        7840
  . LHCURR=4HHEAP                                                       7850
                                                                        7860
HERE,WE PUT OUR CODE FOR GENERATING AN EXTRA LEVEL OF REFERENCING AT    7870
RUN TIME. THE BEGINNING IS SAVED FOR USE AS DESTINATION IN ASSIGNMENTS  7880
                                                                        7890
    CALL OUTCOA5 (Z,10HHEAP+1^HEA,10HP,HEAP+1 ^,10H(HEAP),   8HSAVEU,,  7900
   110HHEAP^U, //)                                                      7910
    RETURN                                                              7920
                                                                        7930
    ENTRY PSI30A2                                                       7940
                                                                        7950
    LHCURR=3HLOC                                                        7960
    CALL OUTCOA3 (Z,10H U^GENERAT,10HOR VALUE, ,10HPOP U,  //)          7970
                                                                        7380
THE VALUE OF THE GENERATOR IS PRESERVED FOR USE IN ASSIGNMENTS9         7990
                                                                        8000
    OUTCODE(Z,3)=10HPOP U,  //                                          8010
    RETURN                                                              8020
    ENTRY PSI31A1                                                       8030
                                                                        8040
        CALL INITIAL(NAME,MODE)
        RETURN
    ENTRY PSI31A2                                                       8050
                                                                        8060
        CALL FINAL(NAME,MODE)
        RETURN
                                                                        8080
    RETURN                                                              8090
                                                                        8100
    ENTRY PSI33A1                                                       8110
                                                                        8120
    ENTRY PSI33B1                                                       8130
                                                                        8140
    LHCURR=3HLOC                                                        8150
        CALL BLOKBGN(P,MODE)
    ITEMP1=Z+3                                                          8210
    K=K+1                                                               8220
    CODELOC(K)=ITEMP1                                                   8230
    K=K+1                                                               8240
    CODELOC(K)=ITEMP1                                                   8250
                                                                        8260
RESERVES SPACE FOR THE TRANSLATION OF INITIALIZATION DECLARATIONS AND   8270
FOR MANAGING RUN-TIME STORAGE ALLOCATION.                               8280
                                                                        8290
    Z=Z+4                                                               8300
    RETURN                                                              8310
                                                                        8320
    ENTRY PSI33A2                                                       8330
                                                                        8340
    ENTRY PSI33B2                                                       8350
                                                                        8360
        CALL BLOKEND(P,MODE)
    RETURN                                                              8430
                                                                        8440
```

```
PSI34B  CAUSES LOCAL BLOCK STORAGE TO BE ERASED IN THE TRANSLATED          8450
PROGRAM.                                                                   8460
                                                                           8470
    ENTRY PSI34A                                                           8480
                                                                           8490
FILLS IN BLANK SPACES AT THE BEGINNING OF TRANSLATED CODE FOR THE          8500
CURRENT BLOCK.                                                             8510
                                                                           8520
    ITEMP=CODELOC(K)-2                                                     8530
    ITEMP1=ITEMP+3                                                         8540
    DO 138 ITEMP2=ITEMP,ITEMP1,1                                           8550
138 OUTCODE(ITEMP2,1)=2H//                                                 8560
    K=K-1                                                                  8570
    RETURN                                                                 8580
                                                                           8590
    ENTRY PSI34B1                                                          8600
                                                                           8610
    K=K-1                                                                  8620
    RETURN                                                                 8630
                                                                           8640
    ENTRY PSI34B2                                                          8650
                                                                           8660
CODE IS PUT OUT AT THIS POINT TO FILL IN THE STORAGE MANIPULATION          8670
BLANKS LEFT BY PSI33A1 AND PSI33B1.                                        8680
                                                                           8690
    ITEMP=CODELOC(K)-2                                                     8700
    CALL OUTCOP2 (ITEMP,10H1+BLOKNUM*,10HBLOKNUM,//)                       8710
    CALL OUTCOB3 (ITEMP,4HLOC*,10HSTORAGE(BL,10HOKNUM), //)                8720
    CALL OUTCOA6 (Z,10HSTORAGE(BL,10HOKNUM)*LOC,1H,10HBLOKNUM-1,10H*BL     8730
   10KNUM, ,10H        //)                                                8740
    RETURN                                                                 8750
                                                                           8760
    ENTRY PSI36A                                                           8770
                                                                           8780
CLEARS THE MOST RECENT DECLARATION FROM THE N STACK AND RESETS CODELOC      8790
FOR THE NEXT DECLARATION.                                                  8800
                                                                           8810
    LHCURR=3HLCC                                                           8820
    CODELOC(K)=Z+1                                                         8830
    Z=Z+2                                                                  8840
    I=I-1                                                                  8850
    RETURN                                                                 8860
                                                                           8870
                                                                           8880
CASE/PSI33A1/<SER. CL.>IN/PSI82D2/<ROW OF CL.>FI/PSI82C3/                  8890
CASE/PSI33A1/<SER. CL.>IN/PSI82D2/<ROW OF CL.>ELSE/PSI82D3/<SER.CL.>FI     8900
                                                                           8910
                                                                           8920
    ENTRY PSI82D2                                                          8930
        IF(.NOT.SAMETYP(N(I),INTREAL,REF))
   1    CALL ERROR(110HINDEX OF A CASE STATEMENT IS NOT A NUMERICAL QU
   2ANTITY.                                                              )
    IF (.NOT.(TYPE(I).EQ.9RNONSTORED)) GO TO 183                           8950
    P=P+1                                                                  8960
    ITEMP=CONVERT(P,2R T)                                                  8970
```

```
      CALL OUTCDA4 (Z,OPERAND(I),1H',ITEMP,10H,       //)                    8980
      CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                                 8990
      I=I-1                                                                 9000
      CODELOC(K)=Z+1                                                        9010
      CALL OUTCDA4 (Z,ITEMP,6H    ≤0',0,10H.;;      //)                    9020
      CALL OUTCDA6 (Z,ITEMP,6H     > ,0,6H      ',0,10H.;;      //)         9030
      CALL OUTCDA6 (Z,ITEMP,6H     + ,0,6H      ',ITEMP,10H,       //)     9040
      CALL OUTCDA3 (Z,ITEMP,10H'RJ(30'47),10H, RJ.,  //)                    9050
                                                                            9060
T(P)≤0'LABEL ELSE.;;                                                        9070
T(P)> NUMBER OF CLAUSES IN ROW' LABEL ELSE.;;                               9080
T(P)+ LABEL FI ' T(P), (SEE PSI82D3 FOR COMPLETION.)                        9090
T(P)'RJ(30'47),RJ.,                                                         9100
CAUSES A JUMP TO A TABLE LEADING TO THE APPROPRIATE  CLAUSE OF THE          9110
CASE STATEMENT, NOTE THAT STORAGE(BLOKNUM) IS USED TO RESET P TO THE        9120
SAME LEVEL FOR EACH CLAUSE.                                                 9130
THE SAVING OF CODELOC(K) IN WHAT FOLLOWS IS PERFORMED FOR COMPATIBILI       9140
WITH PSI33A1 WHEN A<ROW OF CL.> APPEARS WITHIN A <COLLATERAL CLAUSE>.       9150
                                                                            9160
          CALL BLOKEND(P,MODE)
          K=K+1           $CODELOC(K)=Z
      Z=Z+1                                                                 9220
      K=K+1                                                                 9230
      CODELOC(K)=100                                                        9240
      RETURN                                                                9250
                                                                            9260
                                                                            9270
      ENTRY PSI82D3 .                                                       9280
                                                                            9290
SEARCH HERE FOR THE MOST DEREFERENCED AND DEPROCEDURED (I.E.,LOWEST)        9300
CLAUSE IN THE <ROW OF CLAUSE>, AND BRING THE REMAINING CLAUSES DOWN         9310
TO THAT LEVEL.                                                              9320
                                                                            9330
          CALL BLOKEND(P,MODE)
          CALL BLOKBGN(P,MODE)
      K=K-2                                                                 9350
      S=S-1                                                                 9360
      Z=Z-1                                                                 9370
      LINK1=MINIMUM(N(I))                                                   9380
C                                                                           9390
C RULES PSI83A AND PSI83B FOR <ROW OF CLAUSE> CREATE A LIST OF LABELS       9400
C ON TOP OF THE OPERAND STACK.EACH ELT. OF THE LIST CONTAINS IN SEQUENC     9410
C                   (1) A POINTER TO THE NEXT ELT.                          9420
C                   (2) THE LABEL WHERE THE UNITARY CLAUSE BEGINS.          9430
C                   (3) THE LABEL WHERE DEREFERENCING AND DEPROCEDURING     9440
C                       OF THAT CLAUSE OCCURS.                              9450
C                   (4) THE LOCATION IN THE CODE OF (3) WHERE ESCAPE IS     9460
C                       MADE TO THE CODE DIRECTLY FOLLOWING THE CASE        9470
C                       STATEMENT, (SEE BALANCE SUBROUTINE.)                9480
C                                                                           9490
      ITEMP1=OPERAND(I)                                                     9500
      ITEMP2=N(I)                                                           9510
      CALL BALANCE (LINK1,ITEMP1,ITEMP2,S,P,Z,OUTCODE)                      9520
      N(I)=LINK1                                                            9530
C                                                                           9540
```

```
 LINK1 POINTS TO THE LOWEST DATA TYPE IN THE ROW OF CLAUSE.                9550
 SEE  PSI82D2  FOR CONTENTS OF CODELOC(K) AND CODELOC(K-1) HERE.           9560
                                                                           9570
      S=S+1                                                                9580
      ITEMP=CONVERT(S,2R L)                                                9590
      CALL OUTCOA2 (Z,ITEMP,10H:,       //)                              9600
      ITEMP1=CODELOC(K)                                                    9610
      K=K-1                                                                9620
      OUTCODE(ITEMP1+2,3)=ITEMP                                            9630
                                                                           9640
 THE LOCATION OF THE CASE STATEMENT JUMP TABLE IS PUT INTO THE JUMP COD    9650
                                                                           9660
      ITEMP2=OPERAND(I)                                                    9670
 139  CALL OUTCOA2 (Z,CONT(ITEMP2+2),10H.,      //)                       9680
      ITEMP2=LINK(CONT(ITEMP2),1)                                          9690
      IF (ITEMP2.GT.0) GO TO 139                                           9700
                                                                           9710
 THE JUMP TABLE IS GENERATED. THE ELSE CLAUSE LABEL IS NEXT INSERTED.      9720
                                                                           9730
      S=S+1                                                                9740
      ITEMP=CONVERT(S,2R L)                                                9750
      CALL OUTCOA2 (Z,ITEMP,10H:       //)                                9760
      OUTCODE(ITEMP1,3)=ITEMP                                              9770
      OUTCODE(ITEMP1+1,3)=CONVERT(IBYTE(CONT(OPERAND(I)),10),2R  )         9780
      OUTCODE(ITEMP1+1,5)=ITEMP                                            9790
      GO TO 184                                                            9800
 C                                                                         9810
      ENTRY PSI82D4                                                        9820
 C                                                                         9830
 C  FURTHER BALANCING TAKES PLACE HERE BETWEEN THE <ROW OF CLAUSE> AND TH   9840
 C  <SERIAL CLAUSE>.                                                       9850
 C  AFTER SAVING THE BLOCK VALUE, DO THE BLOCK END BOOKKEEPING.            9930
 C                                                                         9860
           CALL BLOKEND(P,MODE)
           P=P+1
      ITEMP4=CONVERT(P,2R T)                                               9890
      IF (OPERAND(I).EQ.ITEMP4) GO TO 140                                  9900
      CALL OUTCOA4 (Z,OPERAND(I),1H@,ITEMP4,10H,       //)                 9910
 140      K=K-1
 C                                                                         10000
 C  TEST HERE FOR WHETHER THE <SER. CL.> IS LOWER THAN THE <ROW OF CL.>.   10010
 C  IF LOWER, JUMP AT RUN TIME TO THE CODE FOLLOWING THE CASE STATEMENT,   10020
 C  CAUSE THE <ROW OF CL.> RESULTING TO BE FURTHER DEREFERENCED AND DEPRO  10030
 C  IF HIGHER, DEREFERENCE AND DEPROCEDURE THE <SER. CL.>; THEN INSERT TH  10040
 C  FOR JUMPING OUT OF THE <ROW OF CL.> ROUTINES.                          10050
 C                                                                         10060
      IF (.NOT.SAMETYP(N(I-1),N(I),REF)) GO TO 141                         10070
      IF (REF.EQ.0) GO TO 142                                              10080
      GO TO 144                                                            10090
 141  IF (SAMETYP(N(I),N(I-1),REF)) GO TO 145                              10100
 C                                                                         10110
           CALL ERROR(110HTHE REMAINING CLAUSE AFTER*ELSE* IN THE CASE ST
     1ATEMENT DOES NOT HAVE THE SAME MODE AS PRECEDING CLAUSES.       )
 C                                                                         10130
      RETURN                                                               10140
```

```
                                                                        10150
 NO BALANCING IS NEEDED HERE.                                           10160
                                                                        10170
142 S=S+1                                                               10180
    ITEMP=CONVERT(S,2R L)                                               10190
    CALL OUTCDA2 (Z,ITEMP,10H:        //)                             10200
    CALL LOOP82D (OPERAND(I-1),ITEMP,OUTCODE)                           10210
                                                                        10220
 PROVIDES ESCAPE LABELS TO THE CASE CLAUSES.                            10230
                                                                        10240
143 I=I-1                                                               10250
    TYPE(I)=6RSTORED                                                    10260
    OPERAND(I)=ITEMP4                                                   10270
    RETURN                                                              10280
                                                                        10290
 THE ENTIRE<ROW OF CL.>,AFTER BEING BALANCED INTERNALLY, IS HERE BALAN  10300
 THE SERIAL  ELSE CLAUSE.                                               10310
                                                                        10320
144 S=S+1                                                               10330
    ITEMP=CONVERT(S,2R L)                                               10340
    CALL OUTCDA2 (Z,ITEMP,10H.,       //)                            10350
    TYPE(I)=ITEMP                                                       10360
    S=S+1                                                               10370
    ITEMP=CONVERT(S,2R L)                                               10380
                                                                        10390
 DO LOOP82D TO FORCE ALL <ROW OF CL.>  TO ESCAPE TO THIS POINT IN THE   10400
 CODE. NOTE THAT VALUE RETURN IS USED TO CARRY THE VALUES OF EACH CLAU  10410
                                                                        10420
    CALL LOOP82D (OPERAND(I-1),ITEMP,OUTCODE)                           10430
                                                                        10440
                                                                        10450
    CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                               10460
    CALL OUTCDA2 (Z,TYPE(I),10H:        //)                          10470
    N(I-1)=N(I)                                                         10480
    GO TO 143                                                           10490
                                                                        10500
THE <SERIAL CL.> MUST BE DEREFERENCED AND DEPROCEDURED TO BALANCE WITH  10510
<ROW OF CL.>.                                                           10520
                                                                        10530
145 S=S+1                                                               10540
    ITEMP=CONVERT(S,2R L)                                               10550
    CALL OUTCDA2 (Z,ITEMP,10H.,       //)                            10560
    CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                               10570
    CALL LOOP82D (OPERAND(I-1),ITEMP,OUTCODE)                           10580
    CALL OUTCDA2 (Z,ITEMP,10H:        //)                            10590
    GO TO 143                                                           10600
                                                                        10610
    ENTRY PSI82C3                                                       10620
        CALL BLOKEND(P,MODE)
    K=K-2                                                               10640
    LINK1=MINIMUM(N(I))                                                 10650
    S=S-1                                                               10660
    Z=Z-1                                                               10670
    CALL BALANCE (LINK1,OPERAND(I),N(I),S,P,Z,OUTCODE)                  10680
    N(I)=LINK1                                                          10690
```

```
      TYPE(I)=6RSTORED                                                    10700
      S=S+1                                                               10710
      ITEMP1=CONVERT(S,2R L)                                              10720
      CALL OUTCOA2 (Z,ITEMP1,10H:,       //)                             10730
      ITEMP=CODELOC(K)                                                    10740
      K=K-1                                                               10750
      OUTCODE(ITEMP+2,3)=ITEMP1                                           10760
      ITEMP3=OPERAND(I)                                                   10770
  146 CALL OUTCOA2 (Z,CONT(ITEMP3+2),10H,,      // )                     10780
      ITEMP3=LINK(CONT(ITEMP3),1)                                         10790
      IF (ITEMP3.GT.0) GO TO 146                                          10800
      S=S+1                                                               10810
      ITEMP1=CONVERT(S,2R L)                                              10820
      CALL OUTCOA2 (Z,ITEMP1,10H:       //)                              10830
      OUTCODE(ITEMP,3)=ITEMP1                                             10840
      ITEMP=ITEMP+1                                                       10850
      OUTCODE(ITEMP,3)=CONVERT(IBYTE(CONT(OPERAND(I)),10),2R  )           10860
      OUTCODE(ITEMP,5)=ITEMP1                                             10870
      P=P+1                                                               10880
      ITEMP=CONVERT(P,2R T)                                               10890
      CALL OUTCOA3 (Z,5H0  ,ITEMP,10H,       //)                         10900
      S=S+1                                                               10910
      ITEMP1=CONVERT(S,2R L)                                              10920
      CALL OUTCOA2 (Z,ITEMP1,10H:       //)                              10930
      CALL LOOP82D (OPERAND(I),ITEMP1,OUTCODE)                            10940
      TYPE(I)=6RSTORED                                                    10950
      OPERAND(I)=ITEMP                                                    10960
      RETURN                                                              10970
                                                                          10980
      IF/PSI33A1/<SERIAL CL.>THEN/PSI82A2/<SER.CL.>ELSE/PSI82A3/<SER.CL.>FI   10990
      IF/PSI33A1/<SERIAL CL.>THEN/PSI82A2/<SER.CL.>FI/PSI82B3/            11000
                                                                          11010
         ENTRY PSI82A2                                                    11020
              IF(.NOT.SAMETYP(N(I),BOOL,REF))
        1     CALL ERROR(110HIF STATEMENT WITH NON-BOOLEAN TEST.
        2                                                        )
         IF (.NOT.(TYPE(I).EQ.9RNONSTORED)) GO TO 147                     11040
         P=P+1                                                            11050
         ITEMP=CONVERT(P,2R T)                                            11060
         CALL OUTCOA4 (Z,OPERAND(I),1H*,ITEMP,10H,       //)              11070
      147 CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                           11080
         I=I-1                                                            11090
         CALL OUTCOA4 (Z,CONVERT(P,2R T),3H=0(,0,10H.,       //)          11100
         CODELOC(K)=Z                                                     11110
;                                                                         11120
;     CODELOC CARRIES LOCATION TO INSERT FIXUP LABEL HERE.                11130
;                                                                         11140
      148      CALL BLOKEND(P,MODE)
               CALL BLOKBGN(P,MODE)
         ITEMP=Z+3                                                        11180
         K=K+1                                                            11190
         CODELOC(K)=ITEMP                                                 11200
         K=K+1                                                            11210
         CODELOC(K)=ITEMP                                                 11220
         Z=ITEMP+1                                                        11230
```

```
      RETURN                                                              11240
C                                                                         11250
      ENTRY PSI82A3                                                       11260
      IF (OPERAND(I).EQ.ITEMP) GO TO 149                                  11270
      ITEMP=CONVERT(IBYTE(STORAGE(BLOKNUM),10)+1,2R T)                    11280
      CALL OUTCDA4 (Z,OPERAND(I),1H=,ITEMP,10H,        //)                11290
      TYPE(I)=6RSTORED                                                    11300
      OPERAND(I)=ITEMP                                                    11310
  149 CALL OUTCDA2 (Z,0,10H.,       //)                                  11320
C                                                                         11330
C  INSERT SKIP CLAUSE LABEL HERE.                                         11340
C                                                                         11350
      K=K-1                                                               11360
      ITEMP1=CODELOC(K)                                                   11370
      CODELOC(K)=7                                                        11380
      S=S+1                                                               11390
      ITEMP=CONVERT(S,2R L)                                               11400
      CALL OUTCDA2 (Z,ITEMP,10H(        //)                              11410
      OUTCODE(ITEMP1,3)=ITEMP                                             11420
      GO TO 148                                                           11430
C                                                                         11440
      ENTRY PSI82A4                                                       11450
          CALL BLOKEND(P,MODE)
          P=P+1
      ITEMP=CONVERT(P,2R T)                                               11470
      IF (OPERAND(I).EQ.ITEMP) GO TO 150                                  11480
      CALL OUTCDA4 (Z,OPERAND(I),1H=,ITEMP,10H,        //)                11490
  150 K=K-1                                                               11500
C                                                                         11510
C  BALANCING OF CONSEQUENCE AND ALTERNATIVE(                              11520
C                                                                         11530
      IF (.NOT.SAMETYP(N(I-1),N(I),REF)) GO TO 151                        11540
      IF (REF.EQ.0) GO TO 152                                             11550
      GO TO 153                                                           11560
  151 IF (SAMETYP(N(I),N(I-1),REF)) GO TO 155                             11570
          CALL ERROR(110HTHE CONSEQUENCE AND ALTERNATIVE OF AN IF STATEM
     1ENT DO NOT HAVE THE SAME A POSTERIORI MODE.                          )
C                                                                         11590
C  NO BALANCING NEEDED.                                                   11600
C                                                                         11610
  152 S=S+1                                                               11620
      ITEMP1=CONVERT(S,2R L)                                              11630
      CALL OUTCDA2 (2,ITEMP1,10H(        //)                             11640
      ITEMP2=CODELOC(K)                                                   11650
      OUTCODE(ITEMP2,1)=ITEMP1                                            11660
      K=K-1                                                               11670
      GO TO 154                                                           11680
C                                                                         11690
C  LOWER THE CONSEQUENCE TO THE LEVEL OF THE ALTERNATIVE,AND CAUSE THE A  11700
C  TIVE TO SKIP OVER THE LOWERING CODE.                                   11710
C                                                                         11720
  153 N(I-1)=N(I)                                                         11730
      CALL OUTCDA2 (2,0,10H.,       //)                                  11740
      ITEMP=7                                                             11750
      S=S+1                                                               11760
```

```
      ITEMP1=CONVERT(S,2R L)                                              11770
      ITEMP2=CODELOC(K)                                                   11780
      K=K-1                                                               11790
      CALL OUTCOA2 (Z,ITEMP1,10H$        //)                             11800
      OUTCODE(ITEMP2,1)=ITEMP1                                            11810
      CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                               11820
      S=S+1                                                               11830
      ITEMP1=CONVERT(S,2R L)                                              11840
      CALL OUTCOA2 (Z,ITEMP1,10H$        //)                             11850
      OUTCODE(ITEMP,1)=ITEMP1                                             11860
  154     I=I-1
      RETURN                                                              11890
C                                                                         11900
C  LOWER THE ALTERNATIVE, THEN INSERT JUMP CODE FOR THE CONSEQUENCE.      11910
C                                                                         11920
  155 CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                               11930
      S=S+1                                                               11940
      ITEMP1=CONVERT(S,2R L)                                              11950
      CALL OUTCOA (Z,ITEMP1,10H$         //)                             11960
      ITEMP2=CODELOC(K)                                                   11970
      K=K-1                                                               11980
      OUTCODE(ITEMP2,1)=ITEMP1                                            11990
      GO TO 154                                                           12000
C                                                                         12010
      ENTRY PSI82B3                                                       12020
C                                                                         12030
C  INSERT ALTERNATIVE CODE AND PROVIDE A JUMP TO AVOID THIS CODE.         12040
C                                                                         12050
      ITEMP=STORAGE(BLOKNUM)                                              12060
      P=IBYTE(ITEMP,10)+1                                                 12070
      ITEMP1=CONVERT(P,2R T)                                              12080
      CALL BLOKEND (0,0)                                                  12090
      CALL BLOKBGN (0,0)                                                  12100
      IF (OPERAND(I).EQ.ITEMP1) GO TO 156                                 12110
      CALL OUTCOA4 (Z,OPERAND(I),1H#,ITEMP1,10H,        //)               12120
      TYPE(I)=6RSTORED                                                    12130
      OPERAND(I)=ITEMP1                                                   12140
  156 CALL OUTCOA2 (Z,0,10H.,       //)                                   12150
      S=S+1                                                               12160
      ITEMP2=CONVERT(S,2R L)                                              12170
      K=K-1                                                               12180
      ITEMP3=CODELOC(K)                                                   12190
      K=K-1                                                               12200
      CALL OUTCOA2 (Z,ITEMP2,10H$        //)                             12210
      OUTCODE(ITEMP3,3)=ITEMP2                                            12220
      CALL OUTCOA3 (7,5H0 #  ,ITEMP1,10H,        //)                     12230
      S=S+1                                                               12240
      ITEMP2=CONVERT(S,2R L)                                              12250
      OUTCODE(7-2,1)=ITEMP2                                               12260
      CALL OUTCOA2 (Z,ITEMP2,10H$        //)                             12270
      RETURN                                                              12280
      ENTRY PSI84A                                                        12290
C                                                                         12300
C  TEST TO SEE WHETHER WE ARE COMPILING THE FIRST CLAUSE IN A ROW OF  CL   12310
C  THEN,IF TRUE,INSERT A JUMP AROUND THE ROW OF CLAUSE TO WHERE THE EVAL  12320
```

```
 JUMP TABLE BEGINS.                                                      12330
                                                                         12340
      P=IBYTE(STORAGE(BLCKNUM),10)                                       12350
      ITEMP=CONVERT(1+P,2R T)                                            12360
      IF (OPERAND(I).EQ.ITEMP) GO TO 157                                 12370
      CALL OUTCOA4 (Z,OPERAND(I),1H*,ITEMP,10H,        //)               12380
  157 CALL OUTCOA3 (Z,10HRETURN* RJ,8H(30*47),,10HRJ.,     //)           12390
      ITEMP1=IFETCH(4)                                                   12400
      IF (CODELOC(K).EQ.-1) GO TO 158                                    12410
                                                                         12420
THIS TEST DISTINGUISHES THE FIRST CLAUSE OF THE ROW.                     12430
                                                                         12440
      K=K+1                                                              12450
      CODELOC(K)=-1                                                      12460
      S=S+1                                                              12470
      ITEMP3=CONVERT(S,2R L)                                             12480
                                                                         12490
 THE DOUBLE USE OF <ROW OF CL.> WITHIN CASE STATEMENTS AND COLLATERAL    12500
 REQUIRES SOME BLANKING OF OUTCODE CARDS PUT IN BY PSI33A1 DURING PSI8   12510
                                                                         12520
      ITEMP2=CODELOC(K-2)                                                12530
      CALL OUTCOA2 (ITEMP2,ITEMP3,10H*       //)                        12540
      CODELOC(K-1)=ITEMP3                                                12550
      CODELOC(K-2)=ITEMP2-1                                              12560
  158 CALL STRIND (CODELOC(K-1),ITEMP1+1)                                12570
      OPERAND(I)=ITEMP1                                                  12580
                                                                         12590
STORED ON OPERAND(I) IN A LIST CELL.                                     12600
                                                                         12610
      S=S+1                                                              12620
      ITEMP3=CONVERT(S,2R L)                                             12630
      CODELOC(K-1)=ITEMP3                                                12640
      CALL OUTCOA2 (Z,ITEMP3,10H*       //)                             12650
      RETURN                                                             12660
                                                                         12670
THE LABEL AT WHICH CODE BEGINS FOR EACH ROW IN A <ROW OF CLAUSE> IS      12680
                                                                         12690
      ENTRY PSI83A                                                       12700
                                                                         12710
 DO PSI84A WORK FIRST,THEN PIECE TOGETHER THE COMPILE-TIME REPRESENTAT   12720
 OF<ROW OF CLAUSE>.                                                      12730
                                                                         12740
      P=IBYTE(STORAGE(BLOKNUM),10)                                       12750
      ITEMP=CONVERT(P+1,2R T)                                            12760
      IF (OPERAND(I).EQ.ITEMP) GO TO 159                                 12770
      CALL OUTCOA4 (Z,OPERAND(I),1H*,ITEMP,10H,        //)               12780
  159 CALL OUTCOA3 (Z,10HRETURN *RJ,8H(30*47),,10HRJ.,     //)           12790
      ITEMP=IFETCH(4)                                                    12800
      CALL STRIND (CODELOC(K-1),ITEMP+1)                                 12810
                                                                         12820
 CONSTRUCT LIST OF <ROW OF CL.> PROCEDURES.                              12830
                                                                         12840
      CALL STRIND (ITEMP,OPERAND(I-1))                                   12850
                                                                         12860
 CONSTRUCT A COMPILE TIME STRUCTURE SKELETON REPRESENTATION OF THE DATA  12870
```

```
      ITEMP=IFETCH(1)                                                    12890
      CALL SETIND (N(I),ITEMP,2)                                         12900
      I=I-1                                                              12910
      ITEMP1=IFETCH(1)                                                   12920
      CALL SETIND (N(I),ITEMP1,2)                                        12930
      CALL SETIND (ITEMP,ITEMP1,1)                                       12940
      CALL PUTIND (2,ITEMP1,10)                                          12950
      N(I)=ITEMP1                                                        12960
      RETURN                                                             12970
                                                                         12980
      ENTRY PSI838                                                       12990
                                                                         13000
      ITEMP=IFETCH(1)                                                    13010
      CALL SETIND (N(I),ITEMP,1)                                         13020
      I=I-1                                                              13030
      CALL SETIND (I(I),ITEMP,2)                                         13040
      N(I)=ITEMP                                                         13050
      CALL PUTIND (1+IBYTE(CONT(N(I+1)),10),ITEMP,10)                    13060
      CALL STRIND (OPERAND(I+1),OPERAND(I))                              13070
      RETURN                                                             13080
      ENTRY PSI85A2                                                      13090
                                                                         13100
INSERT JUMP AROUND <ROW OF CLAUSE> CODE,THEN DO BLOCKEND ROUTINE.        13110
                                                                         13120
      K=K-2                                                              13130
      ITEMP=CODELOC(K)-2                                                 13140
      ITEMP1=ITEMP+1                                                     13150
      DO 160 I1=ITEMP,ITEMP1                                             13160
160   OUTCODE(I1,1)=2H//                                                 13170
      CALL OUTCOB2 (CODELOC(K),CODELOC(K+1),10H.,      //)               13180
      GO TO 184                                                          13190
                                                                         13200
      ENTRY PSI48B2                                                      13210
                                                                         13220
AT THIS POINT, THE MODE OF THE <VIRT. MODE DECLARER> MUST BE EITHER S    13230
OR ARRAY, WITH NO PROC OR REF PREFIXES. IN ADDITION, THE <COLLATERAL     13240
AND ITS A POSTERIORI MODE MUST BE THE SAME LENGTH.                       13250
                                                                         13260
      ITEMP1=CONT(N(I))                                                  13270
      ITEMP2=CONT(N(I-1))                                                13280
      P=P+1                                                              13290
      ITEMP3=IBYTE(ITEMP1,10)                                            13300
      IF (ITEMP3.EQ.IBYTE(ITEMP2,10)) GO TO 161                          13310
      CALL ERROR (110HTHE <COLLATERAL CLAUSE>AND ITS A POSTERIORI MODEAR  13320
     1E NOT THE SAME LENGTH.                                    )        13330
      RETURN                                                             13340
161   ITEMP4=LINK(ITEMP1,3)                                              13350
      IF (ITEMP4.EQ.3RARY) GO TO 162                                     13360
      IF (ITEMP4.EQ.3RSTR) GO TO 164                                     13370
      CALL ERROR (110HTHE A POSTERIORI MODE OF A <MOID CAST> WITH A  <CO  13380
     1LLATERAL CLAUSE> IS NOT ARRAY OR STRUCTURE, AS IT MUST BE.  )      13390
      RETURN                                                             13400
                                                                         13410
AT THIS POINT, WE FIND THE LOWEST MODE IN THE <ROW OF CLAUSE>,CONVERT    13420
```

```
 CIRCULAR LIST RESEMBLING A STRUCTURE, AND CALL THE LOWER SUBROUTINE.     13430
                                                                         13440
152 CALL OUTCOA4 (Z,7HSAVEU, ,LHCURR,10H+1↱U,SAVEU,10H,          //)     13450
                                                                         13460
CREATE A RUN-TIME ARRAY:STEP 1: GENERATE THE DESCRIPTOR.                 13470
                                                                         13480
    ITEMP4=CONVERT(ITEMP3+2,2R  )                                        13490
    CALL OUTCOA6 (Z,LHCURR,1H+,ITEMP4,1H↱,LHCURR,10H,          //)     13500
    CALL OUTCOA3 (Z,10H0100000000,10H0000 ↱ (U),10H,          //)     13510
    CALL OUTCOA3 (Z,10HU+2  ↱ (U),10H(0↱17),U+1,10H↱U,        //)     13520
    CALL OUTCOA3 (Z,10H0300000100,10H0000000001,10H↱(U),      //)     13530
    CALL OUTCOA (Z,CONVERT(ITEMP3,2R  ),10H↱(U)(18↱35,8H),      //)     13540
                                                                         13550
STEP 2: CALL LOWER, AFTER GENERATING CIRCULAR LIST FOR LOWEST MODE.      13560
                                                                         13570
    ITEMP4=MINIMUM(N(I))                                                 13580
    ITEMP5=IFETCH(1)                                                     13590
    CALL SETIND (ITEMP5,ITEMP5,1)                                        13600
    CALL SETIND (ITEMP4,ITEMP5,2)                                        13610
    CALL LOWER (ITEMP3,ITEMP5,N(I),OPERAND(I),Z,P)                       13620
                                                                         13630
STEP 3: COMPLETE WORK ON THE RUN-TIME DATA ELEMENT.                      13640
                                                                         13650
153 ITEMP4=CONVERT(P,2R T)                                               13660
    CALL OUTCOA3 (Z,10HPOPU, U ↱ ,ITEMP4,10H,POPU,  //)                  13670
    I=I-1                                                                13680
    OPERAND(I)=ITEMP3                                                    13690
    RETURN                                                               13700
                                                                         13710
 AT THIS POINT, A STRUCTURE IS GENERATED AT RUN TIME FROM THE ELEMENTS    13720
<COLLATERAL CLAUSE>.                                                     13730
                                                                         13740
164 CALL OUTCOA4 (Z,6HSAVEU,,LHCURR,10H+1 ↱U,SAVE,10HU,          //)     13750
    CALL OUTCOA6 (Z,LHCURR,1H+,CONVERT(ITEMP3,2R  ),1H↱,LHCURR,10H,      13760
   1    //)                                                              13770
    CALL LOWER (ITEMP3,LINK(ITEMP2,2),OPERAND(I),Z,P)                    13780
    GO TO 163                                                            13790
                                                                         13800
                                                                         13810
    ENTRY PSI49A1                                                        13820
                                                                         13830
 THE MODE ASSOCIATED ON THE  N-STACK WITH A <VOID CAST> IS 0. NOTE THA   13840
 THE VALUE OF  P DOES NOT CHANGE HERE BECAUSE OF THE ALGOL 68 CONVENTI   13850
 OF LEAVING THE DESTINATION OF AN ASSIGNATION AS ITS VALUE AND BECAUSE    13860
 OUR PSI33A2 RESETS P.                                                   13870
                                                                         13880
    N(I)=0                                                               13890
    OPERAND(I)=0                                                         13900
    RETURN                                                               13910
                                                                         13920
    ENTRY PSI48A2                                                        13930
                                                                         13940
        P = P + 1                                                        13950
                                                                         13960
    ITEMP=CONVERT(P,2R T)                                                13970
```

```
    IF (.NOT.(ITEMP.EQ.OPERAND(I))) GO TO 165                              13980
    CALL OUTCOA4 (Z,OPERAND(I),1H=,ITEMP,10H,       //)                    13990
165 IF (SAMETYP(N(I-1),N(I),REF)) GO TO 166                                14000
    CALL ERROR (110H THE A PRIORI MODE OF THIS <MODE CAST> DOES NOT MA     14010
   1TCH ITS A POSTERIORI MODE.                                  )          14020
    RETURN                                                                 14030
166 CALL OUTEVAL (P,REF,OUTCODE,7,1000-Z)                                  14040
    I=I-1                                                                  14050
    OPERAND(I)=ITEMP                                                       14060
    RETURN                                                                 14070
                                                                           14080
    ENTRY PSI49A1                                                          14090
                                                                           14100
THE N(I) MODE ASSOCIATED WITH A <VOID CAST> IS 0, AND WILL CAUSE AN E      14110
MESSAGE FROM ANY CALL OF SAMETYP THAT TAKES A ZERO AS ONE OF ITS PARA      14120
                                                                           14130
    P=P-1                                                                  14140
    N(I)=0                                                                 14150
    OPERAND(I)=0                                                           14160
    RETURN                                                                 14170
                                                                           14180
    ENTRY PSI48A2                                                          14190
                                                                           14200
DEREFERENCING AND DEPROCEDURING MAY TAKE PLACE HERE.                       14210
                                                                           14220
    P=P+1                                                                  14230
    ITEMP=CONVERT(P,2R T)                                                  14240
    IF (ITEMP.EQ.OPERAND(I)) GO TO 167                                     14250
    CALL OUTCOA4 (Z,OPERAND(I),1H=,ITEMP,10H,       //)                    14260
167 IF (SAMETYP(N(I),N(I-1),REF)) GO TO 168                                14270
    CALL ERROR (110HA PRIORI MODE OF THE <MODE CAST> DOES NOT MATCHITS     14280
   1 A POSTERIORI MODE.                                         )          14290
    RETURN                                                                 14300
168 CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                                  14310
    I=I-1                                                                  14320
    OPERAND(I)=ITEMP                                                       14330
    RETURN                                                                 14340
                                                                           14350
                                                                           14360
    ENTRY PSI16A                                                           14750
                                                                           14760
THE INTERNAL REPRESENTATION OF REF AND PROC MODE PREFIXES  IS CONSIST      14770
BECAUSE OF LINKING FURTHER PREFIXES BY THE 2ND FIELD.                      14780
                                                                           14790
    CALL PUTIND (1+IBYTE(CONT(N(I)),10),N(I),10)                           14800
    ITEMP1=1                                                               14810
    GO TO 171                                                              14820
                                                                           14830
    ENTRY PSI16B                                                           14840
                                                                           14850
    ITEMP1=2                                                               14860
171 ITEMP=IFETCH(1)                                                        14870
    CALL SETIND (N(I),ITEMP,ITEMP1)                                        14880
    CALL SETIND (3RPRC,ITEMP,3)                                            14890
    N(I)=ITEMP                                                             14900
```

```
      RETURN                                                                14910
C                                                                           14920
      ENTRY PSI15B1                                                         14930
C                                                                           14940
      K=K+1                                                                 14950
      CODELOC(K)=Z                                                          14960
      RETURN                                                                14970
C                                                                           14980
      ENTRY PSI16C                                                          14990
C                                                                           15000
      ITEMP=IFETCH(1)                                                       15010
      CALL SETIND (N(I),ITEMP,2)                                            15020
      CALL SETIND (N(I-1),ITEMP,1)                                          15030
      CALL PUTIND (IBYTE(CONT(N(I-1)),10),N(I),10)                          15040
      I=I-1                                                                 15050
      N(I)=ITEMP                                                            15060
      RETURN                                                                15070
C                                                                           15080
      ENTRY PSI77A1                                                         15090
C                                                                           15100
C  IN THE PROCEDURE CALL, THE <PRIMARY> MUST BE LOWERED TO ITS FIRST MOD    15110
C  A PROCEDURE WITH PARAMETERS.                                             15120
C                                                                           15130
      NI=N(I)                                                               15140
      ITEMP3=LOCF(ITEMP4)                                                   15150
      IF (NI.GT.0) GO TO 173                                                15160
  172 CALL ERROR (110HPROCEDURE CALL ATTEMPTED ON A NON-PROCEDURE <PRIMA    15170
     1RY>.                                                          )       15180
      RETURN                                                                15190
  173 ITEMP=CONT(NI)                                                        15200
      ITEMP1=LINK(ITEMP,3)                                                  15210
C                                                                           15220
C  THE MODE PREFIXES MUST BE SOME SEQUENCE OF REF,STOPPING AT THE FIRST     15230
C  WITH PARAMETERS.                                                         15240
C                                                                           15250
      IF (ITEMP1.NE.3RREF) GO TO 175                                        15260
      ITEMP2=3RREF                                                          15270
  174 ITEMP5=IFETCH(1)                                                      15280
      CALL SETIND (ITEMP2,ITEMP5,3)                                         15290
      CALL SETIND (ITEMP5,ITEMP3,2)                                         15300
      ITEMP3=ITEMP5                                                         15310
      NI=LINK(ITEMP,2)                                                      15320
      GO TO 173                                                             15330
  175 IF (ITEMP1.NE.3RPRC) GO TO 172                                        15340
      IF (LINK(ITEMP,2).GT.0) GO TO 176                                     15350
      ITEMP2=3RPRC                                                          15360
      GO TO 174                                                             15370
  176 REF=ITEMP4                                                            15380
      P=P+1                                                                 15390
      ITEMP4=CONVERT(P,2R T)                                                15400
      IF (OPERAND(I).EQ.ITEMP4) GO TO 177                                   15410
      CALL OUTCDA4 (Z,OPERAND(I),1H*,ITEMP4,10H,        //)                15420
      OPERAND(I)=ITEMP4                                                     15430
      TYPE(I)=6RSTORED                                                      15440
  177 N(I)=NI                                                               15450
```

```
      CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                                  15460
C                                                                            15470
C  NEXT, STORE A LABEL ON N(I) FOR  USE IN SETTING UP THE ACTUAL PARAMET     15480
C  LIST,AND PROVIDE A JUMP AROUND THE ACTUAL PARAMS. LIST  FOR LATER CONF    15490
C                                                                            15500
      I=I+1                                                                  15510
      S=S+1                                                                  15520
      ITEMP4=CONVERT(S,2R L)                                                 15530
      N(I)=ITEMP4                                                            15540
      CALL OUTCOA4 (Z,0,0,ITEMP4,10H1        //)                            15550
      TYPE(I)=Z                                                              15560
      CALL SETDIR (P+1,TYPE(I),2)                                            15570
      RETURN                                                                 15580
C                                                                            15590
      ENTRY PSI76B1                                                          15600
C                                                                            15610
C  THIS RULE CREATES A SEQUENCE OF LIST CELLS CONTAINING MODE INFORMATIO     15620
C  RUN-TIME LABELS LOCATING WHERE EACH <UNITARY CLAUSE> BEGINS.              15630
C                                                                            15640
  178 ITEMP=LINK(TYPE(I-1),2)                                                15650
      ITEMP3=50                                                              15660
      ITEMP1=CONVERT(ITEMP,2R T)                                             15670
      IF (ITEMP1.EQ.OPERAND(I)) GO TO 179                                    15680
      CALL OUTCOA4 (Z,OPERAND(I),1H*,ITEMP1,10H,        //)                   15690
  179 P=ITEMP-1                                                              15700
      CALL OUTCOA3 (Z,10HRETURN* RJ,10H(30*47),  ,10HRJ.,     //)            15710
      ITEMP2=IFETCH(2)                                                       15720
      CALL STRIND (N(I-1),ITEMP2+1)                                          15730
C                                                                            15740
C  STORES LABEL IN SECOND WORD.                                              15750
C                                                                            15760
      CALL SETIND (N(I),ITEMP2,2)                                            15770
C                                                                            15780
C  STORES MODE LINK IN FIRST WORD.                                           15790
C                                                                            15800
      OPERAND(I)=ITEMP2                                                      15810
      IF (ITEMP3.EQ.0) RETURN                                                15820
      S=S+1                                                                  15830
      ITEMP2=CONVERT(S,2R L)                                                 15840
      CALL OUTCOA2 (Z,ITEMP2,10H1        //)                                15850
      N(I)=ITEMP2                                                            15860
      CALL SETDIR (ITEMP,TYPE(I),2)                                          15870
      RETURN                                                                 15880
C                                                                            15890
      ENTRY PSI78A1                                                          15900
C                                                                            15910
C  ABBREVIATED VERSION OF PSI78B1 WITHOUT INSERTION OF THE NEXT LABEL IN     15920
C  OUTPUT CODE.                                                              15930
C                                                                            15940
      ITEMP3=0                                                               15950
      GO TO 178                                                              15960
C                                                                            15970
      ENTRY PSI78B2                                                          15980
C                                                                            15990
C  LINKS TOGETHER THE LIST OF LABELS AND MODE LINKS.                         16000
```

```
C                                                                       16010
      CALL SETIND (OPERAND(I),OPERAND(I-1),1)                           16020
      I=I-1                                                             16030
      RETURN                                                            16040
C                                                                       16050
      ENTRY PSI77A2                                                     16060
C                                                                       16070
C  OPERAND(I) CONTAINS THE LIST OF ENTRY LABELS AND DATA TYPES FOR EACH  16080
C  ACTUAL PARAMETERS. P HAS RESET TO ITS RETURN LEVEL - 1 BY PSI78A1. LH 16090
C  SAVED BEFORE PARAMETER EVALUATION,SO THAT ITS INITIAL VALUE BEFORE ST 16100
C  PARAMETERS ON IT CAN BE RETRIEVED TO PASS  TO THE PROCDEF. AT END OF  16110
C  R EVALUATION, CHECK FOR SAME LENGTH OF P-LISTS. TYPE(I-1) CONTAINS TH 16120
C  TO INSERT A JUMP OVER THE PARAMETERS, AND N(I-2) CONTAINS THE A POSTE 16130
C  PROCEDURE MODE.                                                       16140
C                                                                       16150
      ITEMP=TYPE(I-1).AND.777777B                                       16160
      S=S+1                                                             16170
      ITEMP1=CONVERT(S,2R L)                                            16180
      OUTCODE(ITEMP,1)=ITEMP1                                           16190
      OUTCODE(ITEMP,2)=10H.,                                            16200
      CALL OUTCDA2 (Z,ITEMP1,10H'       //)                           16210
      P=P+1                                                             16220
      NI=CONVERT(P,2R T)                                                16230
      CALL OUTCDA4 (Z,7HSAVE U,,LHCURR,5H+1,U,,10HSAVE U, //)           16240
C                                                                       16250
C  LOWER<ACTUAL PARAMETERS> TO A POSTERIORI LEVEL, THEN STORE IN TURN IN 16260
C                                                                       16270
      ITEMP5=0                                                          16280
C                                                                       16290
C  COUNT OF PARAMETERS                                                  16300
C                                                                       16310
      ITEMP1=OPERAND(I)                                                 16320
      ITEMP3=LINK(CONT(N(I-2)),1)                                       16330
C                                                                       16340
C  ITEMP1 AND ITEMP3 ARE,RESPECTIVELY, LINKS TO THE A PRIORI AND A POSTE 16350
C  PARAMETER LISTS.                                                     16360
C                                                                       16370
  180 ITEMP2=CONT(ITEMP1)                                               16380
      ITEMP4=CONT(ITEMP3)                                               16390
      ITEMP5=1+ITEMP5                                                   16400
      IF (SAMETYP(LINK(ITEMP2,2),LINK(ITEMP4,2),REF)) GO TO 181         16410
          CALL ERROR(110HTHE PARAMETER IN THE JTH POSITION OF THE PROCED
     1URE CALL DOES NOT MATCH ITS A POSTERIORI MODE. J IS...           )
          CALL ERROR1(CONVERT(ITEMP5,2R  ))
      GO TO 180                                                         16450
  181 CALL OUTCDA6 (Z,10HSAVE RETUR,10HN,J+2,RETU,3HRN,,CONT(ITEMP1+1),1 16460
     110H.,POP RETU,10HRN,     //)                                      16470
      CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                             16480
      CALL OUTCDA4 (Z,LHCURR,3H+1,LHCURR,10H,       //)                 16490
      CALL OUTCDA4 (Z,NI,1H,,LHCURR,10H,       //)                     16500
      ITEMP1=LINK(ITEMP2,1)                                             16510
      ITEMP3=LINK(ITEMP4,1)                                             16520
      IF ((ITEMP1.GT.0).AND.(ITEMP3.GT.0)) GO TO 180                    16530
      IF (ITEMP1.EQ.ITEMP3) GO TO 182                                   16540
      CALL ERROR (110HILLEGAL PROCEDURE CALL--THE NUMBER OF ACTUAL AND F 16550
```

```
   10RMAL PARAMETERS DOES NOT MATCH.                                )        16560
    P=P-1                                                                   16570
    I=I-2                                                                   16580
    RETURN                                                                  16590
                                                                            16600
PASS PARAMETERS AT RUN TIME,AND TRANSFER CONTROL TO THE SUBROUTINE.         16610
                                                                            16620
182 CALL OUTCOA4 (Z,8HPOP U,U*,10HPARAMETERS,10H,POP U, //)                 16630
    I=I-2                                                                   16640
    CALL OUTCOA4 (Z,OPERAND(I),8H(18*35)*,9HDEF LEVEL,6H,    //)             16650
                                                                            16660
 BEFORE JUMPING TO THE SUBROUTINE, RAISE PROC LEVEL AND SAVE THE RUN-       16670
 TIME ENVIRONMENT.                                                          16680
                                                                            16690
    CALL OUTCOA2 (Z,10HENVIRONMEN,10HT SAVE, //)                            16700
    CALL OUTCOA6 (Z,10HSAVE RETUR,2HN,,OPERAND(I),10H(0*17)*JUM,10HP,J      16710
   1+2*RETU,10HRN,     //)                                                 16720
    CALL OUTCOA5 (Z,8H(JUMP).,,10HPOPRETURN,,10HRESTORE EN,10HVIRONMEN      16730
   1T,10H,//)                                                               16740
    CALL OUTCOA3 (Z,8HVALUE * ,OPERAND(I),10H,       //)                    16750
    TYPE(I)=6RSTORED                                                        16760
    P=P-1                                                                   16770
    RETURN                                                                  16780
                                                                            16790
    ENTRY PSI73A1                                                           16800
                                                                            16810
 TAKES THE RUN-TIME PROCLEVEL AND THE LABEL AT THE BEGINNING OF <MOID       16820
 AND STORES IT IN A TP REGISTER. RESTORES DEFLEVEL ENVIRONMENT. JUMPS        16830
 ROUTINE COMPLETELY.                                                        16840
                                                                            16850
    P=P+1                                                                   16860
    ITEMP=CONVERT(P,2R T)                                                   16870
    S=S+1                                                                   16880
    ITEMP1=CONVERT(S,2R L)                                                  16890
    CALL OUTCOA5 (Z,1H^,ITEMP1,1H*,ITEMP,10H,       //)                    16900
    CALL OUTCOA4 (Z,10HPROC LEVEL,1H*,ITEMP,10H(18*35),//)                  16910
    CALL OUTCOA6 (Z,0 ,2H.,,ITEMP1,10H:MASK ENVI,10HRONMENT,  10H,//)       16920
                                                                            16930
 THE MASK ENVIRONMENT ROUTINE TESTS DEF LEVEL. IF DEFLEVEL = 0, THEN N      16940
 RESTORATION. OTHERWISE THE  ENVIRONMENT SAVE TABLE IS DUMPED ONTO THE      16950
 NAME TABLE  AT THE LEVEL GIVEN BY DEF LEVEL.                               16960
                                                                            16970
    I=I+1                                                                   16980
    OPERAND(I)=ITEMP                                                        16990
    TYPE(I)=6RSTORED                                                        17000
    N(I)=7                                                                  17010
                                                                            17020
 N(I) SAVES THE LOCATION IN CODE FOR THE OVERJUMP                           17030
                                                                            17040
        CALL PROCRGN(P,0)
                                                                            17060
 ENTERS THE HASH TABLE AND DOES BLOCK BEGIN BOOKKEEPING                     17070
                                                                            17080
    RETURN                                                                  17110
                                                                            17120
```

```
      ENTRY PSI73A2                                                      17130
C                                                                        17140
      ITEMP=IFETCH(1)                                                    17150
      CALL SETIND (3RPRC,ITEMP,3)                                        17160
      CALL SETIND (N(I),ITEMP,2)                                         17170
      CALL OUTCOA4 (Z,OPERAND(I),10H * VALUE, ,10H(RETURN).,,10H         17180
     1//)                                                                17190
      S=S+1                                                              17210
      ITEMP1=CONVERT(S,2R L)                                             17220
      I=I-1                                                              17230
      ITEMP2=N(I)                                                        17240
      OUTCODE(ITEMP2,1)=ITEMP1                                           17250
      CALL OUTCOA2 (Z,ITEMP1,10H)        //)                            17260
      N(I)=ITEMP                                                         17270
          CALL PRCEND(P,0)
      RETURN                                                             17300
C                                                                        17310
C  *************************                                             17320
C                                                                        17330
C                                                                        17390
      END                                                                17400
      SUBROUTINE OUTEVAL (P,REF,OUTCODE,Z,MAX)                             10
      INTEGER P,REF,OUTCODE,Z,CONT,CONVERT                                 20
      DIMENSION OUTCODE(MAX,6)                                             30
      ITEMP=CONVERT(P,2R T)                                                40
  101 ITEMP3=LINK(CONT(REF),2)                                             50
C                                                                          60
C REF POINTS TO THE LIST CREATED BY SAMETYP CONTAINING INFORMATION          70
C ABOUT THE CORRECT SEQUENCE FOR DEPROCEDURING AND DEREFERENCING           80
C THE EXPRESSION CURRENTLY BEING EVALUATED.                                90
C                                                                         100
      IF (ITEMP3.EQ.0) RETURN                                             110
      IF (ITEMP3.EQ.3RREF) GO TO 102                                      120
C                                                                         130
C THE REMAINING ALTERNATIVE, WHICH MUST BE TRUE, IS THAT ITEMP SIGNALS A  140
C PROCEDURE CALL. THE NUMBER OF PARAMETERS IS FOUND IN LINK 3 OF THE      150
C WORD POINTED TO BY REF.                                                 160
C                                                                         170
      ITEMP4=CONVERT(LINK(CONT(REF),3),2R  )                              180
      CALL OUTCOA4 (Z,ITEMP4,10H*NUMBER OF,10HPARAMATERS,8H,      //)      190
      P=P-LINK(CONT(REF),3)                                               200
      ITEMP=CONVERT(P,2R T)                                               210
      CALL OUTCOA4 (Z,10HSAVE RETUR,3HN, ,10HITEMP*RJ( ,10H30*47), //)    220
C                                                                         230
C CONTROL IS PASSED TO THE PROCEDURE9                                     240
C                                                                         250
      CALL OUTCOA2 (Z,10HJ+2*RETURN,10H, RJ*,  //)                        260
      CALL OUTCOA5 (Z,10HPOPRETURN,,10HVALUE RETU,3HRN*,ITEMP,10H,        270
     1 //)                                                                280
      CALL OUTCOA2 (Z,10HPOP VALUE ,10HRETURN, //)                        290
      REF=LINK(CONT(REF),1)                                               300
      GO TO 101                                                           310
C                                                                         320
C THE NUMBER OF REFERENCE LEVELS IS FOUND IN LINK 3 OF THE WORD POINTED   330
C TO BY REF.                                                              340
```

```
C                                                                       350
  102 ITEMP4=LINK(CONT(REF),3)                                          360
      GO TO 104                                                         370
  103 ITEMP4=ITEMP4-1                                                   380
  104 CALL OUTCDA5 (Z,1H(,ITEMP,2H),,ITEMP,10H,        //)              390
      IF (ITEMP4.GT.0) GO TO 103                                        400
      REF=LINK(CONT(REF),1)                                             410
      GO TO 101                                                         420
C                                                                       430
      END                                                               440
      INTEGER FUNCTIONMINIMUM(NI)                                        10
      LOGICAL SAMETYP                                                   100
      INTEGER REF,CONT                                                   90
      ITEMP1=CONT(NI)                                                    20
      LINK1=LINK(ITEMP1,2)                                               30
      ITEMP=LINK(ITEMP1,1)                                               40
  101 ITEMP1=CONT(ITEMP)                                                 50
      LINK2=LINK(ITEMP1,2)                                               60
      IF (SAMETYP(LINK1,LINK2,REF)) GO TO 102                            70
C                                                                       110
C  AT 5, LINK2 IS THE LOWER DATA TYPE.                                  120
C                                                                       130
      IF (SAMETYP(LINK2,LINK1,REF)) GO TO 103                           140
      CALL ERROR (82)                                                   150
      RETURN                                                            160
  102 LINK1=LINK2                                                       170
  103 ITEMP=LINK(ITEMP1,1)                                              180
      IF (ITEMP.GT.0) GO TO 101                                         190
      MINIMUM=LINK1                                                     200
      RETURN                                                            210
C                                                                       220
      END                                                               230
      SUBROUTINE BALANCE (LINK1,ITEMP1,ITEMP2,S,P0,Z,OUTCODE)            10
      DIMENSION OUTCODE(5000,6)                                          30
      LOGICAL SAMETYP                                                   130
      INTEGER S,P,Z,OUTCODE,CONT,P0                                      20
      ITEMP3=LINK(CONT(ITEMP2),2)                                        40
  101 S=S+1                                                              50
      ITEMP=CONVERT(S,2R L)                                              60
      P=P0+1                                                             70
      CALL STRIND (ITEMP,ITEMP1+2)                                       80
      CALL OUTCDA2 (Z,ITEMP,10H(        //)                            90
      CALL OUTCDA3 (Z,10HSAVE RETUR,10HN,J+2*RETU,10RN,      // )        100
      CALL OUTCDA2 (Z,CONT(ITEMP1+1),10H.,       //)                    110
      CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                             150
      CALL STRIND (Z+1,ITEMP1+3)                                        160
C                                                                       170
C  HERE,THE ESCAPE ADDRESS IS PUT INTO THE THIRD LIST ELT. DATUM.       180
C  SEE PSI82D3 FOR DETAILS. NOTE THAT THE ROWS OF CLAUSE USE TICODELOC(K 190
C  AS THE VALUE RETURN ABOVE.                                           200
C                                                                       210
      CALL OUTCDA2 (Z,0,10H.,       //)                                220
      ITEMP1=CONT(ITEMP1)                                               230
      ITEMP2=LINK(CONT(ITEMP2),1)                                       240
      IF (ITEMP1.GT.0) GO TO 101                                        250
```

```
      IF (ITEMP2.EQ.0) RETURN                                              260
           CALL ERROR(110HERROR IN COMPILER,S BALANCE SUBROUTINE,
     1                                                          )
           STOP
      RETURN                                                               280
      END                                                                  300
      SUBROUTINE LOOP820 (LINK,LABEL,OUTCODE)                               10
      DIMENSION OUTCODE(5000,6)                                             20
      INTEGER OUTCODE                                                       30
           ITEMP=INHALT(LINK+3)
  101      OUTCODE(ITEMP,1)=LABEL
      ITEMP=LINK(CONT(ITEMP),1)                                             60
      IF (ITEMP.EQ.0) RETURN                                                70
      GO TO 101                                                             80
C                                                                           90
      END                                                                  100
           SUBROUTINE LOWER(LENGTH,STNDARD,ROW,OPRANDI,Z,P0)
           INTEGER ROW,STNDARD,OPRANDI,Z,P0,P,CONT,REF,CONVERT
      LOGICAL SAMETYP                                                       90
C                                                                           20
C LENGTH GIVES THE NUMBER OF FIELDS TO LOWER, STANDARD IS THE A POSTERIO     30
C STRUCTURE, ROW IS THE A PRIORI MODE STRUCTURE, OPERANDI IS THE LIST OF     40
C CLAUSE> ENTRY LABELS.                                                     50
C                                                                           60
      ITEMP3=CONVERT(P,2R  T)                                               70
           I0=OPRANDI
          I1=STNDARD
      I2=ROW                                                               120
      P=P0                                                                 130
      DO 102 J=1,LENGTH                                                    140
         ITEMP1=CONT(I1)                                                   150
         ITEMP2=CONT(I2)                                                   160
         LINK1=LINK(ITEMP1,2)                                              170
         LINK2=LINK(ITEMP2,2)                                              180
         IF (SAMETYP(LINK2,LINK1,REF)) GO TO 101                           190
          CALL ERROR(110HTHE FOLLOWING <ROW OF CLAUSE> FIELD DOES NOT
     1 MATCH ITS A POSTERIORI MODE.                                        )
          CALL ERROR1(CONVERT(J,2R  )     )
C                                                                          230
C  FIND LABEL OF NEXT CLAUSE IN THE ROW, GENERATE A RETURN JUMP TO IT, T    240
C  CALL OUTEVAL TO LOWER IT.                                               250
C                                                                          260
          CALL OUTCDA3 (Z,10HSAVE RETUR,10HN,J+2HRETU,10HRN,      //)      270
          CALL OUTCDA2 (Z,CONT(I0+1),10H.,       //)                      280
          CALL OUTEVAL (P,REF,OUTCODE,Z,1000-Z)                            290
          CALL OUTCDA3 (Z,6HU+1HU,,ITEMP3,10H U,     //)                   300
  101     I0=CONT(I0)                                                      310
          I1=LINK(ITEMP,1)                                                 320
  102 I2=LINK(ITEMP,2)                                                     330
      RETURN                                                               340
C                                                                          350
      END                                                                  360
           INTEGER FUNCTION IHASH(NAME,MODE)
C                                                                           20
      DIMENSION NAMES1(197,2), NAMES2(197,2), HASH1(197,2), HASH2(197,2)    30
```

```
     1, STORAGE(50), PSTORE(50)                                              40
      INTEGER HASH1,HASH2,STORAGE,BLOKNUM,PRLEVEL,PRCOUNT,BLCOUNT,PSTORE     50
     1,CONVERT                                                               60
      LOGICAL SAMETYP                                                        70
C                                                                            80
      ENTRY INITIAL                                                          90
C                                                                           100
C  CALLED ON ENTRY TO THE COMPILATION OF A NEW PROGRAM.                     110
C                                                                           120
      PRLEVEL=0                                                             130
      BLOKNUM=0                                                             140
      BLCOUNT=0                                                             150
      PRCOUNT=0                                                             160
          MBCOUNT=0
          MPCOUNT=0
      L=IFETCH1(0)                                                          170
C                                                                           180
C  INITIALIZES ALL COMPILE TIME LISTS                                       190
C                                                                           200
      RETURN                                                                210
C                                                                           220
      ENTRY FINAL                                                           230
C                                                                           240
C  CALLED WHEN COMPILER REQUESTS TALLY OF NAMES IN NAMES AND PROCS TABLE    250
C                                                                           260
      NAME=MBCOUNT                                                          270
C                                                                           280
C  MAXIMUM NAMES IN STATIC ENVIRONMENT                                      290
C                                                                           300
          MODE=MPCOUNT
C                                                                           320
C  MAXIMUM NO. OF NAMES IN PROCEDURE ENVIRONMENT.                           330
C                                                                           340
      RETURN                                                                350
C                                                                           360
      ENTRY BLOKBGN                                                         370
C                                                                           380
      BLOKNUM=BLCKNUM+1                                                     390
      STORAGE(BLOKNUM)=BLCOUNT                                              400
          CALL SETDIR(PRCOUNT,STORAGE(BLOKNUM),2)
          CALL SETDIR(NAME,STORAGE(BLOKNUM),3)
  101 L=IFETCH2(0)                                                          410
C                                                                           420
C  COMPILE-TIME LISTS BOOKKEEPING                                           430
C                                                                           440
      RETURN                                                                450
C                                                                           460
      ENTRY BLOKEND                                                         470
C                                                                           480
      L=BLCOUNT-STORAGE(BLOKNUM)-1                                          490
      IF (L.LT.0) GO TO 103                                                 500
      DO 102 I=0,L                                                          510
         J=NAMES1(BLCOUNT-I,1)                                              520
  102 HASH1(J,1)=0                                                          530
  103     BLCOUNT=STORAGE(BLOKNUM).AND.777778
```

```
         PRCOUNT=LINK(STORAGE(BLOKNUM),2)
         NAME=LINK(STORAGE(BLOKNUM),3)
      BLOKNUM=BLCKNUM-1                                                  550
  104 L=IFETCH3(0)                                                       560
;                                                                        570
;  CALLS THE LIST GARBAGE COLLECTER.                                     580
;                                                                        590
      RETURN                                                             600
      ENTRY PROCBGN                                                      610
;                                                                        620
      PRLEVEL=PRLEVEL+1                                                  630
      PSTORE(PRLEVEL)=PRCOUNT                                            640
          CALL SETDIR(NAME,PSTORE(PRLEVEL),2)
      GO TO 101                                                          650
;                                                                        660
      ENTRY PROCEND                                                      670
;                                                                        680
;  DELETES COMPILE TIME NAMES IN CURRENT PROCEDURE BLOCK, BUT LEAVES THE 690
;  TIME LOCATIONS INTACT.                                                700
;                                                                        710
         L=PRCOUNT-1-LINK(PSTORE(PRLEVEL),1)
         NAME=LINK(PSTORE(PRLEVEL),2)
      IF (L.LT.0) GO TO 106                                              730
      DO 105 I=0,L                                                       740
         J=NAMES2(PRCOUNT-I,1)                                           750
  105 HASH2(J,1)=0                                                       760
  106 PRLEVEL=PRLEVEL-1                                                  770
      GO TO 104                                                          780
;                                                                        790
      ENTRY IHASHA                                                       800
;                                                                        810
;  LOOK UP NAME AND MODE,OR LABEL AND LIST OF LOCATIONS TO BE FILLED,AND 820
;  RUN-TIME ALIAS PRECEDED BY ¬.                                         830
;                                                                        840
      LASTUSE=0                                                          850
      I=NAME                                                             860
      IF (PRLEVEL.EQ.0) GO TO 111                                        870
      DO 110 L=0,196,1                                                   880
          I=MOD(I+47*L,197)+1
         IA=HASH2(I,1)                                                   900
         IF (IA.NE.0) GO TO 108                                          910
         IF (LASTUSE.GT.0) GO TO 107                                     920
         GO TO 111                                                       930
  107    IHASHA=CONVERT(LASTUSE,2R¬P)                                    940
         MODE=LOCF(NAMES2(LASTUSE,2))                                    950
         RETURN                                                          960
  108    IF (IA.NE.NAME) GO TO 109                                       970
         LASTUSE=HASH2(I,2)                                              980
  109    CONTINUE                                                        990
  110 CONTINUE                                                          1000
          IF(LASTUSE.GT.0)GO TO 107
C                                                                       1010
C  NAME IS NOT WITHIN PROCEDURE ENVIRONMENT.                            1020
C                                                                       1030
  111 DO 115 L=0,196,1                                                  1040
```

```
          I=MOD(I+47*L,197)+1
         IA=HASH1(I,1)                                                    1060
         IF (IA.NE.0) GO TO 113                                           1070
         IF (LASTUSE.GT.0) GO TO 112                                      1080
          CALL ERROR(110HTHE TAG BELOW IS USED IMPROPERLY9  IT HAS NOT
     19EEN DECLARED IN THIS OR ANY GLOBAL BLOCKS9                    )
          CALL ERROR1(NAME)
  112    IHASHA=CONVERT(LASTUSE,2R^V)                                     1110
         MODE=LOCF(NAMES1(LASTUSE,2))                                     1120
         RETURN                                                           1130
  113    IF (IA.NE.NAME) GO TO 114                                        1140
         LASTUSE=HASH1(I,2)                                               1150
  114    CONTINUE                                                         1160
  115 CONTINUE                                                            1170
          IF(LASTUSE.GT.0)GO TO 112
C                                                                         1180
      ENTRY IHASHB                                                        1190
C                                                                         1200
C  ENTER NAME AND MODE ON NAMES TABLE,AND RETURN RUN-TIME ALIAS PREFIXED  1210
C                                                                         1220
      I=NAME                                                              1230
      IF (PRLEVEL.EQ.0) GO TO 118                                         1240
      DO 116 L=0,196,1                                                    1250
         I=MOD(I+47*L,197)+1                                              1260
         IA=HASH2(I,1)                                                    1270
         IF (IA.EQ.0) GO TO 117                                           1280
C                                                                         1290
C  ENTER THE NAME.                                                        1300
C                                                                         1310
  116 CONTINUE                                                            1320
      CALL ERROR (110HCOMPILE-TIME NAME TABLE OVERFLOW DURING A PROCEDUR  1330
     1E DECLARATION.                                                 )    1340
      RETURN                                                              1350
  117 PRCOUNT=PRCOUNT+1                                                   1360
          IF(PRCOUNT.GT.MPCOUNT)MPCOUNT=PRCOUNT
      HASH2(I,1)=NAME                                                     1370
      HASH2(I,2)=PRCOUNT                                                  1380
      NAMES2(PRCOUNT,1)=I                                                 1390
      NAMES2(PRCOUNT,2)=MODE                                              1400
      MODE=LOCF(NAMES(PRCOUNT,2))                                         1410
C                                                                         1420
C  FOR USE BY LABELS.                                                     1430
C                                                                         1440
      IHASHB=CONVERT(PRCOUNT,2R^P)                                        1450
      RETURN                                                              1460
  118 DO 119 L=0,196,1                                                    1470
         I=MOD(I+47*L,197)+1                                              1480
         IA=HASH1(I,1)                                                    1490
         IF (IA.EQ.0) GO TO 120                                           1500
  119 CONTINUE                                                            1510
      CALL ERROR (110HCOMPILE-TIME NAME TABLE OVERFLOW OUTSIDE OF A  PRO  1520
     1CEDURE DECLARATION.                                            )    1530
      RETURN                                                              1540
  120 BLCOUNT=BLCOUNT+1                                                   1550
      IF (BLCOUNT.GT.MBCOUNT) MBCOUNT=BLCOUNT                             1560
```

```
      HASH1(I,1)=NAME                                                    1570
      HASH1(I,2)=BLCOUNT                                                 1580
      NAMES1(BLCOUNT,1)=I                                                1590
      NAMES1(BLCOUNT,2)=MODE                                             1600
      MODE=LOCF(NAMES1(BLCOUNT,2))                                       1610
      IHASHP=CONVERT(BLCOUNT,2R^V)                                       1620
      RETURN                                                             1630
                                                                         1640
      ENTRY IHASHC                                                       1650
                                                                         1660
C     THIS FUNCTION LOOKS UP OPERATORS IN THE NAME TABLE. WE ASSUME THAT AN   1670
C     OPERATOR NOT PRESENT ON THE TABLE IS EITHER INVALID OR WIRED IN TO TH   1680
C     RUN-TIME PILOT SYSTEM.                                                  1690
                                                                         1700
      LASTUSE=0                                                          1710
      I=NAME                                                             1720
      IF (PPLEVEL.LO.2) GO TO 124                                        1730
      DO 123 L=0,196,1                                                   1740
         I=MOD(I+47*L,197)+1                                             1750
         IA=HASH2(I,1)                                                   1760
         IF (IA.NE.0) GO TO 121                                          1770
         IF (LASTUSE.GT.0) GO TO 107                                     1780
         GO TO 124                                                       1790
  121    IB=HASH2(I,2)                                                   1800
                                                                         1810
C     COMPARE OPERAND MODES.                                             1820
C                                                                        1830
         IB=LINK(CONT(NAMES2(IB,2)),1)                                   1840
         IF ((IA.NE.NAME).AND.(.NOT.SAMETYP(MODE,IB,REF))) GO TO 122     1850
         LASTUSE=HASH2(I,2)                                              1860
  122    CONTINUE                                                        1870
  123 CONTINUE                                                           1880
          IF(LASTUSE.GT.0)GO TO 107
  124 DO 126 L=0,196,1                                                   1890
         I=MOD(I+47*L,197)+1                                             1900
         IA=HASH1(I,1)                                                   1910
         IF (IA.NE.0) GO TO 125                                          1920
         IF (LASTUSE.GT.0) GO TO 112                                     1930
         IHASHC=0                                                        1940
         RETURN                                                          1950
  125    IB=HASH1(I,2)                                                   1960
         IB=LINK(CONT(NAMES1(IB,2)),1)                                   1970
         IF ((IA.NE.NAME).AND.(.NOT.SAMETYP(MODE,IB,REF))) GO TO 126     1980
         LASTUSE=HASH1(I,2)                                              1990
  125 CONTINUE                                                           2000
C                                                                        2010
          IF(LASTUSE.GT.0)GO TO 112
          RETURN
      END                                                                2020
```